NIP - An Imperfection Processor to Data Mining datasets

José M. Cadenas*, M. Carmen Garrido, Raquel Martínez

Department of Engineering Information and Communications, University of Murcia, Faculty of Computer Science. Campus of Espinardo, 30100 Espinardo, Murcia, Spain

E-mail: {jcadenas, carmengarrido, raquel.m.e}@um.es

Received 30 November 2012

Accepted 28 February 2013

Abstract

Every day there are more techniques that can work with low quality data. As a result, issues related to data quality have become more crucial and have consumed a majority of the time and budget of data mining projects. One problem for researchers is the lack of low quality data in order to test their techniques with this data type. Also, as far as we know, there is no software tool focused on the create/manage low quality datasets which treats, in the widest possible way, the low quality data and helps us to create repositories with low quality datasets for testing and comparison of data mining techniques and algorithms. For this reason, we present in this paper a software tool which can create/manage low quality datasets. Among other things, the tool can transform a dataset by adding low quality data, removing and replacing any data, constructing a fuzzy partition of the attributes, etc. It also allows different input/output formats of the dataset.

Keywords: Low quality data, imprecise/uncertain data, dataset with low quality data, Soft Computing, software tool for Soft Computing

1. Introduction

A new challenge in the area of Soft Computing is how to deal with the inherent imperfection of real world information. Although many data mining techniques[1] are based on the assumption of perfect information, in reality data are never as good as engineers would like. Very often, data suffer damage which affects their interpretation. If data mining techniques do not take into account this imperfection in the information, the models generated are low quality defective or unnecessarily complex models. Finally, this affects the interpretation and decision-making that we take based on these data.

There are a wide range of data mining techniques based on different theoretical proposals[2,3,4]. Unfortunately, most of the conventional techniques did not consider the sources of imperfection. As a result, incomplete and imprecise data have usually been discarded or ignored in their learning phases and in their subsequent inference processes. However, these data are inevitable when dealing with real world applications. Also techniques to deal with certain types of low quality data are beginning to be developed[5,6,7].

In this situation, we can analyze different alternatives. We could transform low quality data, which a technique can not manage, by other kinds of values trying not to lose too much information in this change. In this case, we need a software tool to be

*Corresponding author.

able to make this transformation. Another alternative is to modify the technique in order that it can work with low quality data, but in this case the problem is to find public datasets to check the technique, so that we know how a technique behaves when it deals with different levels of imperfection. Besides we could need to introduce expert information expressed by linguistic labels or to discretize certain values by a fuzzy partition to make the dataset interpretable. In these latter situations, we also need a software tool to be able to create datasets with the desired features.

In this paper, we present a new tool, "NIP Imperfection Processor" [a] (hereafter NIP), to manage low quality datasets. Moreover, this software tool allows us to define different formats for input/output datasets both predefined and custom by user.

This paper is organized as follows. In Section 2 we present different kinds of low quality data that this software tool can manage. In Section 3 we outline a brief study about some software tools that can process datasets. Also, in Section 4 we explain the main functionalities of the software tool NIP. Finally, in Section 5 we present several case studies and the conclusions of this paper in Section 6.

2. Low Quality Data

Low quality information inevitably appears in realistic domains and situations. Instrument errors or corruption from noise during experiments may give rise to information with incomplete data when measuring a specific attribute. In other cases, the extraction of exact information may be excessively costly or unfeasible. Moreover, it might be useful to complement the available data with additional information from an expert, which is usually elicited by imperfect data (interval data, fuzzy concepts, etc). In most real world problems, data have a certain degree of imperfection. Due to this situation, datasets from which to extract knowledge or model systems will contain low quality information.

In this section we develop a brief study about some kind of low quality data and how techniques handle them.

[a]Available on the authors website[8]

2.1. Missing Values

An important factor to consider when working with data mining techniques is the treatment that they perform on missing values. Namely:

- Allow missing values in the dataset when the technique is robust to the existence of such values.
- Remove attributes with missing values of the dataset.
- Remove examples with missing values of the dataset.
- Replace missing values manually (if not many) or automatically by a value that preserves the mean or variance (global or classes/groups), in the case of numerical values, or by the mode in the case of nominal values. Another way to estimate a value is to predict it from other examples (missing value imputation) using any predictive data mining technique.
- Replace missing values by an interval that covers the entire domain of the corresponding attributes.

2.2. Noise

Noise is a broad term that has been interpreted in different ways. For instance, it has been defined as a random error in a measured attribute. Another definition states that noise is any property in the detected pattern that is not due to real underlying model but the randomness in the world or the sensors[9]. While a description of the complete taxonomy of data noise is an open research issue, there are generally two types of noise in a given dataset[10]: class noise and attribute noise. Class noise or labeling errors occur when an example belongs to the incorrect class. Class noise can be attributed to several causes, including subjectivity during the labeling process, data entry errors, or the absence of some representative attributes. In contrast, attribute noise reflects erroneous values for one or more attributes (independent attribute) of the dataset. The complexity of detecting class noise is relatively lower than that of detecting attribute noise.

In these definitions, noise is considered an error that occurs randomly. Noise in the data can happen for several reasons:

- First, due to problems with measuring instruments or equipment.
- Second, it is due to the fact that large datasets are obtained by automated methods.

Depending on the data mining technique, we can deal with noise using the following options:

- If it is possible to detect the values with noise, the treatment may be similar to the case of missing values.
- It is sometimes possible to know the errors of the measuring instrument (mean and standard deviation). This allows us to incorporate this information in the set of values of an attribute with noise by means of some transformation.

2.3. Imprecision/Uncertainty

Two kinds of low quality information can be found in our environment and our thinking process: imprecision and uncertainty. Imprecision and uncertainty can be considered as two complementary aspects of imperfect information[11,12,13].

From a practical viewpoint, an item of information can be represented as a quadruple (attribute, object, value, confidence). The attribute is a function which assigns a value (or a set of values) to the object. The value is a subset of the reference domain associated to the attribute. Confidence indicates the likelihood of the item of information.

In this context, imprecision is related to the value of the item, while uncertainty is related to the confidence in the item. Thus, an item of information will be precise when its value cannot be subdivided. Otherwise, we will talk about imprecision. Furthermore, when there are no crisp constraints on the set of values that an imprecise item can take, we will talk about fuzzy imprecision. On the other hand, uncertainty is a property of belief.

We say that we are certain of an event if we assign it a maximum belief value. We can define uncertainty as the absence of certainty, and this may arise from the randomness of some experimenting (objective uncertainty), or from subjective judgments by human reasoning (subjective uncertainty).

In Dubois et al.[14] the concepts of imprecision and uncertainty are described in terms of stochastic and epistemic uncertainties. Stochastic uncertainty arises from random variability related to natural processes. Epistemic uncertainty arises from the imprecise/incomplete nature of available information. While stochastic uncertainty is adequately addressed using classical probability theory, several uncertainty theories have been developed in order to explicitly handle imprecise/incomplete information that basically are convex probability sets, random sets and possibility theory.

Some simple representations of imprecise/uncertain information based on intervals and its generalization are:

- Using an interval $[a,b]$, so we assume that the attribute value is within it.
- Using a fuzzy set that assigns a degree of possibility between 0 and 1 to each value of the interval as a possible attribute value.

Another way of representing the imprecision/uncertainty is by means of crisp or fuzzy subsets of a finite set of values:

- In the case of nominal attributes, using a crisp subset of the domain values, such as $\{$red,blue$\}$ if the domain is $\{$red,blue,green$\}$ or by a fuzzy subset assigning degrees of possibility to a subset of possible attribute values $\{$0.2/red,0.8/blue$\}$.
- In the case of numerical attributes, using a subset of linguistic labels that are associated with a fuzzy partition. For instance, a temperature value of 45 degrees may be transformed to fuzzy subset $\{$0.0/cold,0.2/warm,0.8/hot$\}$. If the partition is a Ruspini's partition, the sum of the membership degrees is 1. In any other type of fuzzy partition, the sum can take another value, such as $\{$0.1/cold,0.3/warm,0.9/hot$\}$.

Depending on how we carry out the treatment of imprecision/uncertainty in different data mining techniques, we can:

- Allow those values in the dataset when the technique is robust to the existence of such kind of values.

- Remove attributes with imprecision/uncertainty of the dataset.
- Remove examples with imprecision/uncertainty of the dataset.
- Replace/impute values with imprecision/uncertainty of the dataset.

3. Software to Pre-Process Data

In this section we revise the main characteristics of some data mining and learning software tools from the viewpoint of data preprocessing. We only focus on exposing the main features of free software, although some private software tools which perform data preprocessing are Knowledge Seeker[15], Simulink Design Optimization[16], etc.

- Sodas2[17]: is a tool that supports symbolic analysis of data. It tries to generalize the data mining and statistical process in a higher level, described by symbolic data. In this way, data is transformed into more manageable and more complex data. The running of Sodas2 allows us to build a set of symbolic data that summarizes the information of an initial dataset, and after this, to perform the symbolic analysis.
  The use of symbolic data allows us to introduce different types of low quality data: multi-valued attributes, intervals and multi-valued attributes with weights. These types of attributes represent other types of low quality data, such as fuzzy data. However, Sodas2 does not allow:
  a. the direct use of fuzzy technology;
  b. the introduction of missing values, noise;
  c. the modification of data to introduce any kind of imperfection.
- Weka[4]: Provides a set of techniques for data preprocessing, classification, regression, clustering, association rules and visualization. For the preprocessing of data, Weka offers a wide variety of techniques to pre-process attributes and examples. In the case of low quality data, it reduces the number of techniques, offering less opportunities.
  Weka is only able to handle missing values, and provides techniques to replace these values by the mode/media. It also allows the addition of noise in nominal attributes.
- Keel[18,19]: It is a tool for data mining software. It contains a large collection of classical techniques for knowledge extraction, pre-processing and learning based on computational intelligence, hybrid models, and a module of statistical tests for comparison.
  Keel allows the use of different input and output formats such as CSV, XML or ARFF. The data management module includes a wide variety of techniques for selection of examples, attributes selection, discretization, etc. Once again, there are few techniques for the management of low quality data, as it only allows the use of missing values by various imputation methods (clearing of examples with nearest k-neighbor imputation, k-means imputation, etc).
- Rapid miner[20]: Formerly called Yale, is an environment for computational learning and data mining paradigm intended to support rapid prototyping. It offers the possibility of using different formats for input/output. The number of preprocessing techniques offered by this tool is great but, again, decreases if we focus on techniques that manage low quality data.
  Rapid miner can manage missing values using multiple imputation methods: replacement with the mean, mode, maximum, minimum or the value predicted by data mining technique defined by the user. It also allows the introduction of noise in numerical attributes and/or nominal.
- MiningMart[21]: It is developed with the purpose of reusing techniques successfully in the preprocessing of large datasets. MiningMart is not focused on the knowledge discovery process, it is only the pre-processor. It offers various aggregation techniques, discretization, data cleaning, treatment of missing values and selection of relevant attributes.
  The only allowed kind of low quality data this tool can manage is missing values, allowing us to delete the examples containing them or replacing them by the mode, media, etc or by an imputation method.

Table 1: Comparative of Characteristics of Software Tools with regard to data management

(Y: Yes; N: No; P: Partially)		Sodas2	Weka	Keel	Rapid Miner	MiningMart	NIP
Format:	Standard	P	Y	Y	Y	P	Y
	Custom	N	N	N	N	N	Y
Low quality data:	missing values	N	Y	Y	P	N	Y
	interval values	N	N	N	N	N	Y
	fuzzy values	N	N	N	N	N	Y
	crisp partition	Y	Y	Y	Y	Y	Y
	fuzzy partition	N	N	N	N	N	Y
	crisp subsets	N	N	N	N	N	Y
	fuzzy subsets	N	N	N	N	N	Y
Imputation/Replacement:	missing values	Y	Y	Y	P	P	P
	interval value	N	N	N	N	N	P
	fuzzy value	N	N	N	N	N	P
Noise in Attributes:	Nominal	N	Y	N	Y	N	Y
	Numerical	N	N	N	Y	N	Y
Attribute Selection		Y	Y	Y	Y	Y	N
Example Selection		Y	Y	Y	Y	Y	N

There are other tools that focus on the process of knowledge extraction, Adam[22], D2K[23], Knime[24], Orange[25] or Tanagra[26], among others, but do not put too much attention to the treatment of low quality data.

Table 1 shows a summary where we can see the main features of the software tools that we have previously presented. In this table we write: "Y" when the tool has completely developed the feature which is described in the corresponding row, "P" when the feature is only partially developed, and "N" when the tool has not developed the feature.

4. NIP: A tool to manage low quality data

4.1. Function modules and main features

NIP is a software tool to manage and generate datasets with low quality data. The main purpose of this software tool is to serve as a research tool to be used in investigation (due to the absence of similar software) and to allow us to establish a common framework for this kind of data. The version of NIP[8] that is presently available consists of three function modules. The tool operates in terms of function modules as follows:

- Import Module: This module is made up of a set of function elements for import and an initial preprocess of the dataset. NIP can import datasets with a standard format (ARFF, KEEL, UCI, CSV) or with a custom format defined by the user. The initial preprocess involves the removal of attributes, the change of attribute type, the ordering of attributes, normalization of numerical attributes, etc. Additionally, this module includes the elements of extraction of information from the dataset (number of examples, attributes, types of attributes, mean, mode, percentages of low quality data contained in the import dataset, etc.).
- Low quality data management Module: The aim of this module is adding different types of low quality data. In numerical attributes, NIP allows us to add data from missing values to fuzzy values (defined by the user or by means of automatic attribute partitioning techniques). In nominal attributes, NIP can add from missing values until crisp/fuzzy subsets. In this module, an internal library of attribute partitioning techniques is included.
- Export Module: This module is one of the fun-

damental parts of NIP. The useful output datasets for the user are constructed in this module. This part is made up of the function elements for export and replacement/imputation of the dataset. As in the import module, the exported dataset will have a standard format or custom format defined by the user. Furthermore, an internal library of replacement/imputation methods is included. With these methods we can generate different samples of a low quality dataset. In addition, NIP enable the export of the dataset by means of a k-fold cross validation. As additional information to the exported dataset, a file which specifies the whole description of the dataset is constructed.

An outline of the process followed by NIP through different modules is shown in Figure 1.

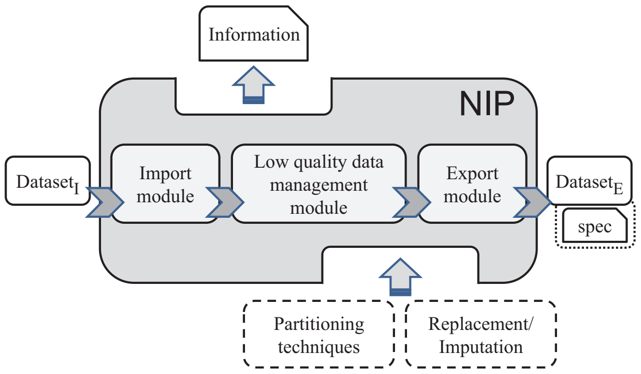


Fig. 1. NIP software tool to manage low quality data

The user will require the availability of a different set of features in order to be interested in using NIP. Below, we describe the main features of NIP:

- An user-friendly interface is provided, oriented towards data management and the construction of low quality datasets.
- Datasets can be imported in standard formats (ARFF, KEEL, UCI, CSV) and in custom format defined by the user.
- Imported datasets can contain low quality data.
- The tool allows us to remove attributes, normalize attributes, change the type of attributes, etc.
- Datasets can be managed to add low quality data of different types (missing, fuzzy sets, interval, crisp/fuzzy subsets, noise, etc).
- The tool contains an internal library of partitioning methods for numerical attributes.
- Useful and descriptive information about the imported and exported datasets is provided.
- The tool contains an internal library of reemplacement/imputation methods of low quality data.
- Datasets can be exported in standard formats (ARFF, KEEL, UCI, CSV) and in custom format defined by the user.
- A file with descriptive information is associated to each exported dataset.
- Multiple output datasets can be created in different formats or sampled datasets by means of replacement/imputation of low quality data in the datasets.
- Datasets can be exported by means of a k-fold cross validation

Since the most differentiating feature about functionality of NIP over other software tools (see Table 1) is the management of low quality data, Figure 2 shows the kinds of low quality data which can be managed by NIP in the current version. This Figure 2 highlights what can be done with each one: read (import module), write (export module), add (management module) and replace/impute (export module).

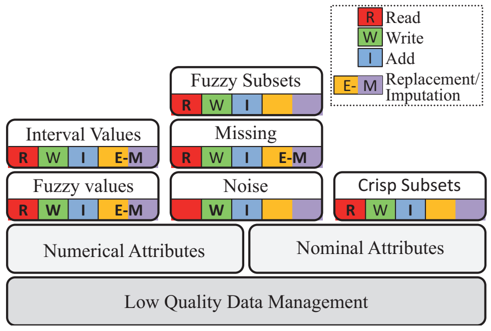


Fig. 2. Low quality data and their management by NIP

The different function modules, which have been presented, are always reflected on the tool. The track of these modules is shown in Figure 3.

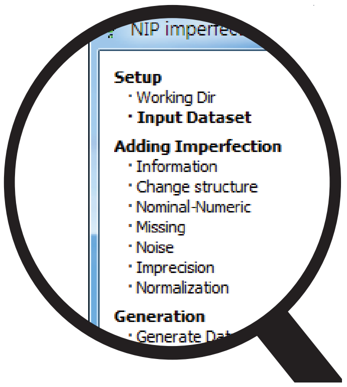


Fig. 3. Tracking process

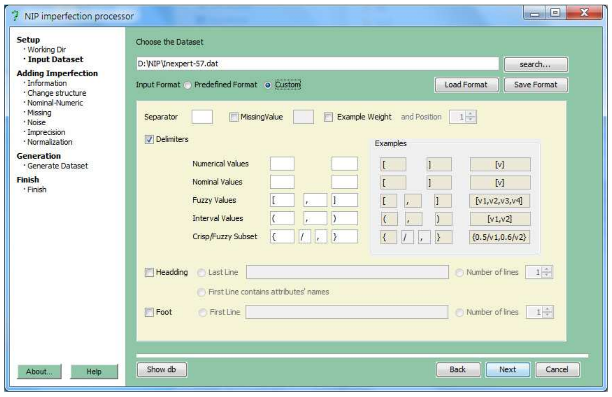
We have briefly presented the function modules of NIP. In the following subsections, we will describe in more detail the possibilities that NIP offers and the methodologies followed by them.

4.2. *Import module*

The main function of this module is to import datasets with which the user wants to work. In the general functioning scheme of this module, two cases are distinguished:

1) A standard format ARFF, KEEL, UCI, CSV will be used when the imported dataset is described in one of these input formats and this dataset only includes types of low quality data allowed by the data mining techniques which use such formats.
2) A custom format will be used when the dataset has a standard input format, but this dataset includes types of low quality data that techniques which use such formats do not allow, or when the dataset has a particular format. Figure 4 shows the screen of definition of a custom format.

In the first case, the methodology of the tool is clear, performing a simple parse internally according to the selected format. By contrast, in the second case, although an internal parse is also done, it requires the cooperation of the user through the user-friendly interface of the tool.



Choose the Dataset

D:\NIP\Inexpert-57.dat

Input Format ( ) Predefined Format [✓ Custom]

Separator ☐ MissingValue ☐ Example Weight and Position 1

☑ Delimiters

				Examples				
Numerical Values				[			]	[v]
Nominal Values				[			]	[v]
Fuzzy Values	[	,		]	[	,		]	[v1,v2,v3,v4]
Interval Values	(	,		)	(	,		)	[v1,v2]
Crisp/Fuzzy Subset	{	/	,	}	{	/	,	}	{0.5/v1,0.6/v2}

☐ Headding ( ) Last Line ( ) Number of lines 1

( ) First Line contains attributes' names

☐ Foot ( ) First Line ( ) Number of lines 1

Fig. 4. Custom input format

Using this interface (Figure 4) the user has to indicate how the different low quality data are described in the dataset:

- A missing value must be represented by a character or string of characters.
- An interval value must be described by the minimum and maximum of the interval.
- A fuzzy value must be represented by the four values that describe its associated membership function.
- A crisp subset is represented by different nominal values that do not have an associated membership degree.
- A fuzzy subset is represented by a set of nominal values that have an associated membership degree.

When the dataset is imported, all information about it is calculated. This information can be divided into: general features of the dataset, statistical information and percentages of low quality data.

The first information collected after importing the dataset is about the features of the dataset, such as the number of attributes, number of examples and nature of attributes (nominal or numerical). These features are necessary to calculate the statistical information since this depends on the nature of the dataset attributes.

For numerical attributes, the statistical information obtained is composed of the mean and the minimum and maximum value. The mean of an attribute may be expressed by a crisp value, an interval value or a fuzzy value. It will be a crisp value when all examples are crisp values for such attribute. In this case the standard deviation is also provided as statistical information. The mean will take an interval value when all values are crisp but there is at least one interval value for that attribute. Finally, it will be a fuzzy value when at least there is a fuzzy value for that attribute. The mean is calculated according to its arithmetic method as discussed above.

For nominal attributes, the statistical information is the mode and all values of the attribute domain along with the proportion of each value in the dataset. Since a nominal attribute may contain crisp values and/or crisp/fuzzy subsets, the calculation of these proportions is performed as follows: for each domain value, its proportion is obtained taking into account the membership degrees of that value in all examples of the dataset for that attribute. For instance, for the value $\{0.1/a, 0.3/b, 0.6/c\}$ a membership degree of 0.1 is considered to the proportion of value a, of 0.3 for the value b and of 0.6 for the value c.

Finally, if the imported dataset has low quality values, the corresponding percentages are calculated. So, for each attribute in the dataset the percentages of fuzzy, interval and missing values and crisp/fuzzy subsets are calculated.

The last feature included in this module is a possible preprocess of the dataset if necessary. This preprocess is mainly focused on:

- Change the nature of an attribute, which can change from nominal to numerical (when possible) or from numerical to nominal.
- Change the order of attributes in the dataset. It can also be possible to eliminate those which are not considered necessary.
- Normalize numerical attributes whose domains are not between 0 and 1.

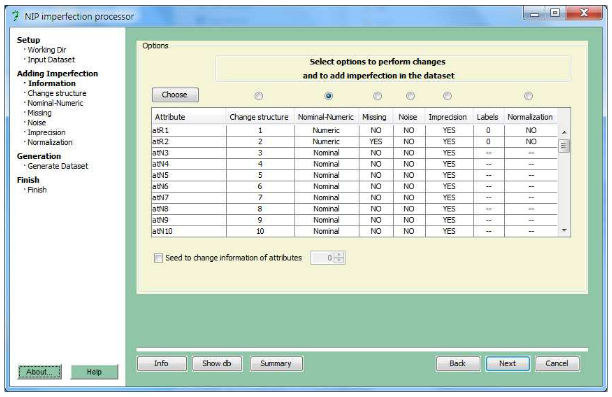
4.3. *Low quality data Management Module*

One of the more important aspects of this tool is to be able to manage low quality values in datasets. This functionality allows the generation of synthetic experiments which enable us to measure the robustness of different data mining techniques from low quality data. With NIP we try to generate dataset repositories that help in this area of research (some low quality datasets are available in the authors website[8]).

In this subsection, we focus on the main functionalities that NIP offers to manage low quality data in datasets. The different types of low quality data allowed are:

- Imprecise values: interval values, fuzzy values and fuzzy subsets in numerical attributes; fuzzy subsets and crisp subsets in nominal attributes.
- Values with noise: gaussian noise in numerical attributes and random noise in nominal attributes.
- Missing values: in both numerical and nominal attributes.

Figure 5 shows the screen of the tool which allows us to access the different options to add low quality data.



Options

Select options to perform changes
and to add imperfection in the dataset

Attribute	Change structure	Nominal-Numeric	Missing	Noise	Imprecision	Labels	Normalization
atR1	1	Numeric	NO	NO	YES	0	NO
atR2	2	Numeric	YES	NO	YES	0	NO
atN3	3	Nominal	NO	NO	YES	--	--
atN4	4	Nominal	NO	NO	YES	--	--
atN5	5	Nominal	NO	NO	YES	--	--
atN6	6	Nominal	NO	NO	YES	--	--
atN7	7	Nominal	NO	NO	YES	--	--
atN8	8	Nominal	NO	NO	YES	--	--
atN9	9	Nominal	NO	NO	YES	--	--
atN10	10	Nominal	NO	NO	YES	--	--

☐ Seed to change information of attributes 0

Fig. 5. Screen of summarized information

In that screen and throughout the modification process of the dataset, the available information about it and the percentages of low quality data which affect different attributes, can be obtained.

When we add low quality data in a dataset, it is always possible to select both the attribute or attributes to modify and the percentage of values of each one of them (except to add gaussian noise to an attribute because the noise affects 100% of its crisp values). This tool controls the total imperfection percentage at all times so that the affect on the same attribute never exceeds 100%. If this happens the tool informs the user of this error condition. Furthermore, since the imported dataset can include low quality data, the corresponding percentages are also take into account in calculating the percentage of total imperfection. For instance, a dataset with 50% of intervals in an attribute, when it is imported by the tool, this dataset can only admit an additional 50% of low quality data in that attribute.

In certain options to add low quality data, a numerical attribute partition is required. In this case, the tool allows us to read that partition from a file provided by the user or to run some algorithm of automatic partitioning included in the internal library contains in the tool.

The first option allows us to use a partition which can be contributed through an expert (in this case we introduce expert information in the dataset) or by means an automatic partitioning algorithm other than those provided by the tool. The second one allows us to create an automatic crisp/fuzzy partition from one of the algorithms that are available in the tool. Currently, the available algorithms are: *OFP_CLASS*[27], which carries out a fuzzy partition using a fuzzy decision tree and a genetic algorithm, a version of the Yoon-Seok Choice's algorithm[28], which constructs a fuzzy partition using a genetic algorithm and a crisp partitioning method defined by phase 1 of the method proposed in Cadenas et. al.[27]. The latter method is based on fuzzy decision trees.

Next we are going to describe more specifically the different functionalities available in the tool to add low quality data to datasets.

4.3.1. Imprecise values

As we have explained in previous sections, possibility and interval theories are used to represent imprecise information. NIP can add imprecise information to datasets in several ways which are described in detail below.

1) Changing crisp values of a numerical attribute by a fuzzy value. In this case, we need to have a fuzzy partition of the attribute that we want to change. Once we have a fuzzy partition, the determined percentage of crisp values will be changed for the fuzzy value of the partition that they belong with to the greatest degree.

2) Changing crisp values of a numerical attribute by an interval value which contains the value of the attribute. In this case the tool provide three possibilities. Firstly, we could generate an interval by adding and subtracting a fixed amount to the value that we want to modify (this option preserves the average value of the attribute). Another possibility is to generate an interval by adding and subtracting a random amount under the maximum value specified in the option. In both cases, the values that form the obtained intervals will be adjusted to the limits of the attribute domain. The latter option is to add interval values from an automatic crisp partition using the internal library of partitioning mentioned above.

3) Replacing a crisp numerical value by a fuzzy subset of linguistic labels of an attribute partition. Given a fuzzy partition of that attribute, the numerical value is replaced by the subset of the partition labels to which this value belongs with degree greater than 0 along with their membership degrees. For instance, a crisp value of 45 for the numerical attribute "temperature" could be replaced by the fuzzy subset $\{0.1/\text{warm}, 1/\text{hot}\}$ if the value 45 belongs to "warm" label with degree 0.1 and to "hot" label with 1 degree. Therefore, to add this kind of low quality data to the dataset a fuzzy partition of numerical attributes is required.

4) Replacing a crisp nominal value by a crisp/fuzzy subset of the domain values of that attribute. If the nominal value is replaced by a crisp subset, all other possible domain values are added to the current value with a probability equal to the proportion of each value in the dataset. If the value is replaced by a fuzzy subset, as membership degree of the current value its proportion in the dataset is added and other values are added in the same way as in the case of crisp subsets including in this case together with the value, its proportion in the dataset as membership degree.

Also, in this NIP's module, we can create a fuzzy or crisp partition of the numerical attributes without introducing imprecision in the dataset. Figure 6 shows an example of a screen for adding low quality data.

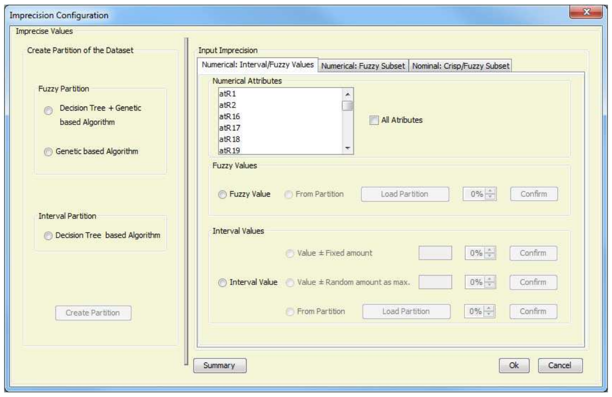


Fig. 6. Adding Low Quality Data

4.3.2. *Values with noise*

We can introduce values with noise, both in nominal and in numerical attributes.

In the case of numerical attributes, the noise that we can add is Gaussian noise. In this kind of noise, it's necessary to indicate for such attribute, the mean and the standard deviation that simulates the error committed to measure attribute values. After specifying these values, the noise would affect the 100% of crisp values of the attribute so it will not be possible to add to this attribute another kind of low quality values.

In the case of the nominal attributes NIP follows the scheme of adding noise used by some authors[29]. For this it's only necessary to express the percentage of noise to indicate the number of values of that attribute that will be changed randomly by another value domain. In order to introduce a noise level of $x\%$ in attribute A, the $x\%$ of the examples in the dataset are chosen approximately and the value of A of each of these examples is assigned a random value belonging to that attribute domain, following an uniform distribution. The percentage of noise in the attribute may be lower than desired, because random assignment can choose the original value again sometimes.

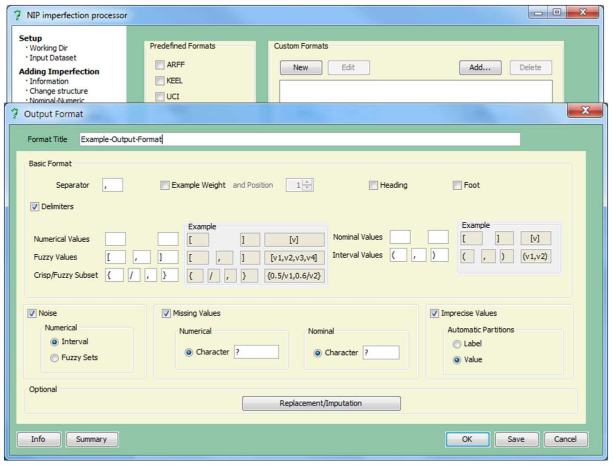
4.3.3. *Missing Values*

Another possibility of adding low quality data in the dataset is by means of missing values. The adding of missing values is random following an uniform distribution.

4.4. *Export Module*

After adding low quality data to the dataset, we are going to work with the export module of the tool whose functionality is described in this subsection.

The main function of the export module is to export the dataset in the desired format. Multiple outputs are allowed, each one with a different format and with the possibility for each one to generate a partition for cross-validation. In the latter case, it has generated the complete dataset and the partitioned one into its corresponding test and train files by means of a k-cross-validation, where k is a configurable parameter. All resulting files are organized in a directory structure created by the tool from the directory selected by the user. Different export formats available in the tool are standard formats ARFF, KEEL, UCI and CSV or a custom format defined by the user. Figure 7 shows a screen of an output format.



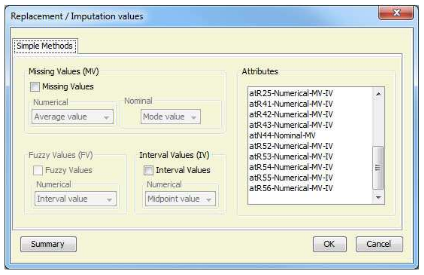
Format Title: Example-Output-Format
Separator: ,
[✓] Delimiters
Numerical Values example: [ ] [v]
Fuzzy Values: [,] — example [v1,v2,v3,v4]
Crisp/Fuzzy Subset: { / , } — example {0.5/v1,0.6/v2}
Nominal Values example: [ ] [v]
Interval Values: (,) — example (v1,v2)
[✓] Noise — Numerical: [✓ Interval] Fuzzy Sets
[✓] Missing Values — Numerical: Character ? — Nominal: Character ?
[✓] Imprecise Values — Automatic Partitions: Label [✓ Value]

Fig. 7. Output format screen

An important and common aspect in the definition of any output format is that fuzzy values and/or intervals, which have been added by a partition, can be represented in the output dataset through their associated linguistic labels or by values that define them.

When the selected format is a standard format, it is possible that the dataset to export contains certain low quality values with which the data mining technique that uses this standard format can not work. These values may have been added by the low quality management module or they have already existed in the original dataset. In this case the user has two alternatives. The first one is to specify a format for these low quality values using NIP if the data mining technique has been adapted to handle these types of data. The second one is to replace/impute low quality values by others appropriate to the data mining technique. This last option solves the possible conflict of low quality values in the standard formats and allows us to work with a dataset that has low quality data originally in the data mining technique.

To carry out the action of replacing/imputing low quality data, NIP provides an internal library of methods of replacement/imputation (Figure 8).



Numerical: Average value
Nominal: Mode value
Fuzzy Values (FV) Numerical: Interval value
Interval Values (IV) Numerical: Midpoint value

atR25-Numerical-MV-IV
atR41-Numerical-MV-IV
atR42-Numerical-MV-IV
atR43-Numerical-MV-IV
atN44-Nominal-MV
atR52-Numerical-MV-IV
atR53-Numerical-MV-IV
atR54-Numerical-MV-IV
atR55-Numerical-MV-IV
atR56-Numerical-MV-IV

Fig. 8. Replacement/imputation of values

This library has several simple methods frequently used in literature to replace/impute low quality values by crisp values[30,31]. Among the methods provided by the tool we can find:

- replace/impute missing values of a numerical attribute by the mean of the attribute, by the mean of the attribute taking its class value as reference, by an interval that includes the whole domain of the attribute or by a fuzzy value constructed as $(\mu - 2\sigma, \mu - \sigma, \mu + \sigma, \mu + 2\sigma)$ where μ and σ are the mean and standard deviation of the attribute. The latter imputation can only be used when the attribute only takes crisp values in all known examples.
- replace/impute missing values of a nominal attribute for mode, a fuzzy subset consisting of the values of the domain along with their proportion, or a crisp subset consisting of all values of the attribute domain.
- replace/impute the fuzzy values through its center of gravity or by an interval which includes all the support of the associated membership function.
- replace/impute the interval values for the midpoint, the minimum or maximum value of the interval.

NIP provides the flexibility to select different choices of replacing/imputing in the same output format allowing to make a chain of replacements/imputations. This chain is performed in the following precedence order: firstly missing values are replaced/imputed, then fuzzy values are replaced/imputed and finally, interval values are replaced/imputed. For instance, if we want to replace/impute a fuzzy value by the midpoint of the interval that defines its support, we will select the option replace/impute fuzzy values by intervals and the option replace/impute intervals by its midpoint.

It is important to notice that when using the replacement/imputation internal library, different samples of the low quality values can be generated in different datasets.

Finally, all the available information in the user-friendly interface of the tool is exported to a file. In the appropriate directory, a file with extension .spec is generated with each selected output format. This file contains information about the exported dataset indicating for each attribute some statistical information, the percentages of low quality data that have been applied on it and if some methods of replacement/imputation have been used.

5. Case studies

In this section, we show two application cases of NIP. To carry out these cases, we have selected the datasets "Inexpert-57.dat" available in the KEEL[19] tool website and "hepatitis.arff" in ARFF format available in the UCI repository[32].

5.1. *Case 1: Inexpert-57.dat dataset*

Inexpert-57.dat[19,33] is a dataset for unsupervised learning with low quality data. It contains data of schoolchildren with dyslexia. It has been estimated that between 4% and 5% of these schoolchildren have this disorder. Dyslexia can be defined as a learning disability in people with a normal intellectual coefficient, and without further physical or psychological problems that can explain such a disability.

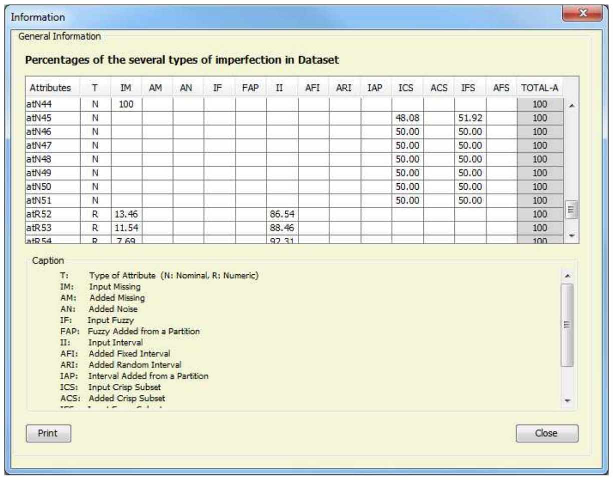
The presence of dyslexic children depends on many different indicators, which are mainly intended to detect whether reading, writing and calculus skills are acquired at the proper rate. Moreover, there are different disorders from dyslexia that share some of their symptoms and therefore the tests not only have to detect abnormal values of the mentioned indicators but in addition, must also separate those children that actually suffer from dyslexia from those where the problem can be related to other causes (inattention, hyperactivity, etc.).

After defining in NIP a custom format suitable to the description of the dataset, we can import it. This dataset is composed of 57 attributes and 52 examples. The attributes are nominal and numerical, and have the following features: attributes 1, 2, 16-25, 41-43 and 52-56 are numerical and defined by intervals. All of them contain missing values except the attribute 1. All others are nominal, defined by crisp/fuzzy subsets. These nominal attributes do not contain missing values. Figure 9 shows some examples of the dataset.

```
cionado,0.4/Regular},{Regular},{Regular},{?},[5.1,7.3],{?},[2,4],[1.3,3.4],{?},[2,3.6],{?
lado},{Basico},{Desproporcionado},{Regular},{Regular},[5.1,7.3],[8,10],[4,6],[4,6],[4,
roporcionado},{0.4/Regular,0.6/Bajo},{Incompleto},{?},[5.1,7.3],{?},{?},[4,6],{?},[0,2]
roporcionado},{Regular},{Regular},{?},[6,8],[6,8],[2,3.4],[2.9,4.1],{?},[1.3,3.4],{?},[1.3
8/Regular,0.4/Basico},{Desproporcionado},{0.7/Bajo,0.4/Regular},{Incompleto},[5.2,7.
,{Basico},{Elevado},{Normal},{Regular},{Regular},[1.3,3.4],[6.0,7.5],[6.9,8.5],[2.9,4.6
{Bajo},{Desproporcionado},{Regular},{Incompleto},[1,2],{?},[3.4,5.1],[1.3,2.6],[1.3,2.
Detallado,Aceptable},{Normal},[4,6],[8,10],[4,6],[4.4,6.3],[4.4,6.3],{?},[7.3,9],{?},{?},{
,0.5/Regular},{Normal},{Aceptable},{Normal},[4,6],{?},[5.8,7.9],[5.5,7.6],[5.1,6.9],{?}
ble},{Normal},[3.4,5.1],[3.8,6],[6.3,7.9],[3.8,5.1],[3,4.7],[8,10],[8,9.7],{?},[1.3,3.4],{?},
sproporcionado},{Regular},{0.82/Regular,0.3/Bajo},[0.2,2],{?},[4.6,6.9],[2.2,4],[1.2,2.5].
{0.4/Regular,0.7/Bajo},{Bajo},[1.3,3.4],{?},{?},[0,0],{?},{?},{?},[3.4,5.1],[0,2],{?},{Im
egular,0.4/Elevado},{Desproporcionado},{0.5/Regular,0.9/Aceptable},{Regular},[2,4],[
},{0.7/Completo,0.3/Normal},[3.4,5.1],[6,8],[7.3,9],[6,8],[3.4,5.1],{?},[7.3,9],{?},[6,8],{?
roporcionado},{Aceptable},{Normal},[4,6],[4,6],[6,8],[2,3.8],[4,5.3],{?},[5.1,7.3],{?},[4
evado,0.4/Basico},{Normal},{Aceptable},{Normal},[2,4],[4,6],[7.3,9],[3.1,4.9],[4,6],{?}
oporcionado},{0.5/Regular,0.5/Elevado},{Regular},[4,6],{?},[6,8],[4,5.8],[3.1,5.0],{?},[1
gular},[4,6],{?},[6,8],[4,6],[3.4,5.1],{?},[4.4,6.1],{?},[2.2,4.3],[4.8,6.5],{Regular},{Si},{
lar,Bajo},{Basico,Elevado,Detallado,Incoherente,Regular,Bajo},{Incoherente,Despropor
Normal,Regular},{Regular},{Regular,Bajo},[5.1,6.9],{?},[6,8],[4,6],[2,4],{?},[1.2,2.4],{?
```

Fig. 9. A part of the "Inexpert-57.dat" dataset

This information is obtained by NIP. In the screen of Figure 10, a part of the information about the amount of imperfection contained in the dataset is shown.



Percentages of the several types of imperfection in Dataset

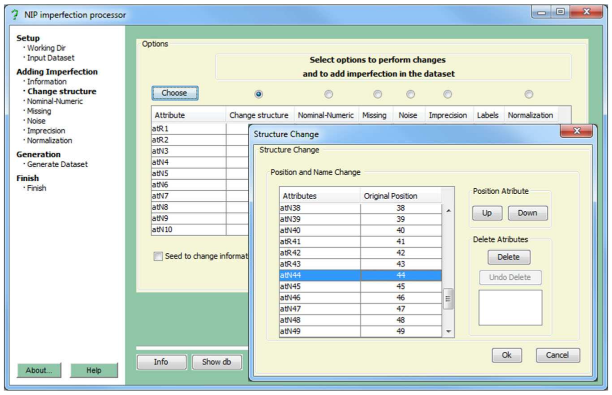
Attributes	T	IM	AM	AN	IF	FAP	II	AFI	ARI	IAP	ICS	ACS	IFS	AFS	TOTAL-A
atN44	N	100													100
atN45	N										48.08		51.92		100
atN46	N										50.00		50.00		100
atN47	N										50.00		50.00		100
atN48	N										50.00		50.00		100
atN49	N										50.00		50.00		100
atN50	N										50.00		50.00		100
atN51	N										50.00		50.00		100
atR52	R	13.46					86.54								100
atR53	R	11.54					88.46								100
atR54	R	7.69					92.31								100

Caption

T:	Type of Attribute (N: Nominal, R: Numeric)
IM:	Input Missing
AM:	Added Missing
AN:	Added Noise
IF:	Input Fuzzy
FAP:	Fuzzy Added from a Partition
II:	Input Interval
AFI:	Added Fixed Interval
ARI:	Added Random Interval
IAP:	Interval Added from a Partition
ICS:	Input Crisp Subset
ACS:	Added Crisp Subset

Fig. 10. Summarized imperfection

As we can see in this figure, the attribute 44 contains 100% of missing values. In this situation we decided to remove it using the previous preprocess option to delete attributes (in the screen of Figure 11 this option is shown).



Attributes	Original Position
atN38	38
atN39	39
atN40	40
atR41	41
atR42	42
atR43	43
atN44	44
atN45	45
atN46	46
atN47	47
atN48	48
atN49	49

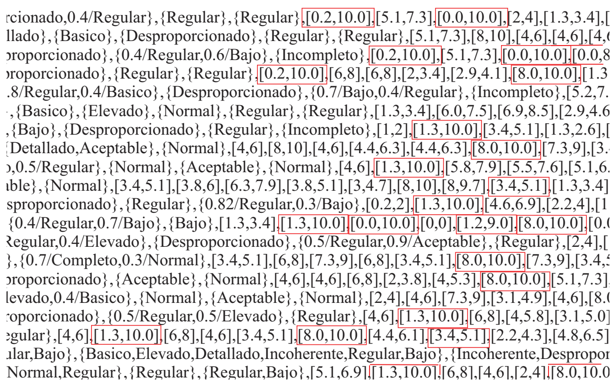
Fig. 11. Removing attribute 44

At this time, we decide to obtain another dataset in which there will not be missing values. To this end, in the export module we define the output format and indicate that we will perform "Replacement/Imputation". Using this functionality, we will replace missing values by interval values. In this situation we consider two options:

- replace them by the mean of that attribute, or
- replace them by an interval representing the whole attribute domain to which the missing value belongs. In this case, a missing value is interpreted as absence of information about its possible value.

For instance, the attribute domain 2 is the interval [0,10] and its mean value is the interval [5.4229,7.3549]. Therefore, in the first option, a missing value would be replaced by the interval [5.4229,7.3549] and in the second one by the interval [0,10].

We choose option 2 and Figure 12 shows part of the transformed dataset. In this figure, we highlight some of the transformed values with respect to the original dataset in Figure 9.



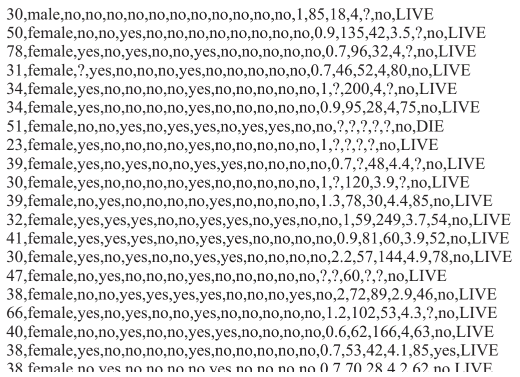
```
rcionado,0.4/Regular},{Regular},{Regular},[0.2,10.0],[5.1,7.3],[0.0,10.0],[2,4],[1.3,3.4],[
llado},{Basico},{Desproporcionado},{Regular},{Regular},[5.1,7.3],[8,10],[4,6],[4,6],[4,
roporcionado},{0.4/Regular,0.6/Bajo},{Incompleto},[0.2,10.0],[5.1,7.3],[0.0,10.0],[0.0,8
roporcionado},{Regular},{Regular},[0.2,10.0],[6,8],[6,8],[2,3.4],[2.9,4.1],[8.0,10.0],[1.3
.8/Regular,0.4/Basico},{Desproporcionado},{0.7/Bajo,0.4/Regular},{Incompleto},[5.2,7.
,{Basico},{Elevado},{Normal},{Regular},{Regular},[1.3,3.4],[6.0,7.5],[6.9,8.5],[2.9,4.6
,{Bajo},{Desproporcionado},{Regular},{Incompleto},[1,2],[1.3,10.0],[3.4,5.1],[1.3,2.6],[
Detallado,Aceptable},{Normal},[4,6],[8,10],[4,6],[4.4,6.3],[4.4,6.3],[8.0,10.0],[7.3,9],[3.
,0.5/Regular},{Normal},{Aceptable},{Normal},[4,6],[1.3,10.0],[5.8,7.9],[5.5,7.6],[5.1,6.
ble},{Normal},[3.4,5.1],[3.8,6],[6.3,7.9],[3.8,5.1],[3,4.7],[8,10],[8,9.7],[3.4,5.1],[1.3,3.4]
sproporcionado},{Regular},{0.82/Regular,0.3/Bajo},[0.2,2],[1.3,10.0],[4.6,6.9],[2.2,4],[1
{0.4/Regular,0.7/Bajo},{Bajo},[1.3,3.4],[1.3,10.0],[0.0,10.0],[0,0],[1.2,9.0],[8.0,10.0],[0.
egular,0.4/Elevado},{Desproporcionado},{0.5/Regular,0.9/Aceptable},{Regular},[2,4],[
},{0.7/Completo,0.3/Normal},[3.4,5.1],[6,8],[7.3,9],[6,8],[3.4,5.1],[8.0,10.0],[7.3,9],[3.4,
roporcionado},{Aceptable},{Normal},[4,6],[4,6],[6,8],[2,3.8],[4,5.3],[8.0,10.0],[5.1,7.3]
levado,0.4/Basico},{Normal},{Aceptable},{Normal},[2,4],[4,6],[7.3,9],[3.1,4.9],[4,6],[8.
oporcionado},{0.5/Regular,0.5/Elevado},{Regular},[4,6],[1.3,10.0],[6,8],[4,5.8],[3.1,5.0]
gular},[4,6],[1.3,10.0],[6,8],[4,6],[3.4,5.1],[8.0,10.0],[4.4,6.1],[3.4,5.1],[2.2,4.3],[4.8,6.5]
lar,Bajo},{Basico,Elevado,Detallado,Incoherente,Regular,Bajo},{Incoherente,Despropor
Normal,Regular},{Regular},{Regular,Bajo},[5.1,6.9],[1.3,10.0],[6,8],[4,6],[2,4],[8.0,10.0
```

Fig. 12. A part of transformed "Inexpert-57.dat" dataset

The output of NIP is the following: 1) Inexpert-57.custom that contains the low quality dataset, and 2) the specification file Inexpert-57.custom.spec.

5.2. Case 2: Hepatitis.arff dataset

Hepatitis.arff is a dataset for supervised learning. It contains medical information about life expectancy of patients with hepatitis. Choosing in NIP the ARFF format we can import it. This dataset is composed of 20 attributes including the class attribute (the latter) and 155 examples. The attributes are nominal, integer and attributes 14 and 17 are real. All of them have missing values except attributes 1, 2, 4, 19 and the class attribute. Figure 13 shows some examples of the dataset.



```
30,male,no,no,no,no,no,no,no,no,no,no,no,1,85,18,4,?,no,LIVE
50,female,no,no,yes,no,no,no,no,no,no,no,no,0.9,135,42,3.5,?,no,LIVE
78,female,yes,no,yes,no,no,yes,no,no,no,no,no,0.7,96,32,4,?,no,LIVE
31,female,?,yes,no,no,no,yes,no,no,no,no,no,0.7,46,52,4,80,no,LIVE
34,female,yes,no,no,no,no,yes,no,no,no,no,no,1,?,200,4,?,no,LIVE
34,female,yes,no,no,no,no,yes,no,no,no,no,no,0.9,95,28,4,75,no,LIVE
51,female,no,no,yes,no,yes,yes,no,yes,yes,no,no,?,?,?,?,?,no,DIE
23,female,yes,no,no,no,no,yes,no,no,no,no,no,1,?,?,?,?,no,LIVE
39,female,yes,no,yes,no,no,yes,yes,no,no,no,no,0.7,?,48,4.4,?,no,LIVE
30,female,yes,no,no,no,no,yes,no,no,no,no,no,1,?,120,3.9,?,no,LIVE
39,female,no,yes,no,no,no,no,yes,no,no,no,no,1.3,78,30,4.4,85,no,LIVE
32,female,yes,yes,yes,no,no,yes,yes,no,yes,no,no,1,59,249,3.7,54,no,LIVE
41,female,yes,yes,yes,no,no,yes,yes,no,no,no,no,0.9,81,60,3.9,52,no,LIVE
30,female,yes,no,yes,no,no,yes,yes,no,no,no,no,2.2,57,144,4.9,78,no,LIVE
47,female,no,yes,no,no,no,yes,no,no,no,no,no,?,?,60,?,?,no,LIVE
38,female,no,no,yes,yes,yes,yes,no,no,no,yes,no,2,72,89,2.9,46,no,LIVE
66,female,yes,no,yes,no,no,yes,no,no,no,no,no,1.2,102,53,4.3,?,no,LIVE
40,female,no,no,yes,no,no,yes,yes,no,no,no,no,0.6,62,166,4,63,no,LIVE
38,female,yes,no,no,no,no,yes,no,no,no,no,no,0.7,53,42,4.1,85,yes,LIVE
38,female,no,yes,no,no,no,no,yes,no,no,no,no,0.7,70,28,4.2,62,no,LIVE
```

Fig. 13. A part of the "hepatitis.arff" dataset

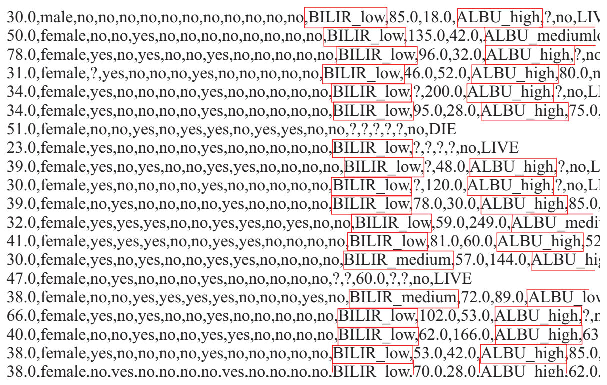
Suppose we want to add labels from an automatic partition in two real attributes. To this end, we use a partition generated by the first fuzzy partitioning algorithm[27] provided in the internal library of NIP (Figure 6). The generated partition is saved in a file. In this file, to each attribute the first line shows the name of the attribute followed by the number of fuzzy sets generated for such attribute. In the next lines of file, the different fuzzy sets generated are shown. Figure 14 shows the fuzzy partition of the attributes.

```
BILIRUBIN 3
        BILIR_low,0.3,0.3,1.29099,2.00863
        BILIR_medium,1.29099,2.00863,2.15801,2.74167
        BILIR_high,2.15801,2.74167,8.0,8.0
ALBUMIN 4
        ALBU_low,2.1,2.1,3.26057,3.6394
        ALBU_mediumlow,3.26057,3.6394,3.8397799,3.8599901
        ALBU_mediumhigh,3.8397799,3.8599901,3.8849301,3.91503
        ALBU_high,3.8849301,3.91503,6.4,6.4
```

Fig. 14. Fuzzy partition of the attributes 14 and 17

Using this partition we obtain a new dataset with all values of attributes 14 and 17 as fuzzy values (expressed by linguistic labels). As we explained in subsection 4.3, each value is replaced by the fuzzy value of the partition that it belongs with to the greatest degree. We may also choose to replace only a percentage of values.

Finally, we obtain the transformed dataset in the selected output format. Figure 15 shows a part of the transformed dataset in ARFF format. In this figure, we highlight some of the transformed values with respect to the original dataset in Figure 13.



```
30.0,male,no,no,no,no,no,no,no,no,no,no,no,BILIR_low,85.0,18.0,ALBU_high,?,no,LIV
50.0,female,no,no,yes,no,no,no,no,no,no,no,no,BILIR_low,135.0,42.0,ALBU_mediuml
78.0,female,yes,no,yes,no,no,yes,no,no,no,no,no,BILIR_low,96.0,32.0,ALBU_high,?,n
31.0,female,?,yes,no,no,no,yes,no,no,no,no,no,BILIR_low,46.0,52.0,ALBU_high,80.0,n
34.0,female,yes,no,no,no,no,yes,no,no,no,no,no,BILIR_low,?,200.0,ALBU_high,?,no,L
34.0,female,yes,no,no,no,no,yes,no,no,no,no,no,BILIR_low,95.0,28.0,ALBU_high,75.0,
51.0,female,no,no,yes,no,yes,yes,no,yes,yes,no,no,?,?,?,?,?,no,DIE
23.0,female,yes,no,no,no,no,yes,no,no,no,no,no,BILIR_low,?,?,?,?,no,LIVE
39.0,female,yes,no,yes,no,no,yes,yes,no,no,no,no,BILIR_low,?,48.0,ALBU_high,?,no,L
30.0,female,yes,no,no,no,no,yes,no,no,no,no,no,BILIR_low,?,120.0,ALBU_high,?,no,L
39.0,female,no,yes,no,no,no,no,yes,no,no,no,no,BILIR_low,78.0,30.0,ALBU_high,85.0,
32.0,female,yes,yes,yes,no,no,yes,yes,no,yes,no,no,BILIR_low,59.0,249.0,ALBU_medi
41.0,female,yes,yes,yes,no,no,yes,yes,no,no,no,no,BILIR_low,81.0,60.0,ALBU_high,52
30.0,female,yes,no,yes,no,no,yes,yes,no,no,no,no,BILIR_medium,57.0,144.0,ALBU_hi
47.0,female,no,yes,no,no,no,yes,no,no,no,no,no,?,?,60.0,?,?,no,LIVE
38.0,female,no,no,yes,yes,yes,yes,no,no,no,yes,no,BILIR_medium,72.0,89.0,ALBU_lo
66.0,female,yes,no,yes,no,no,yes,no,no,no,no,no,BILIR_low,102.0,53.0,ALBU_high,?,n
40.0,female,no,no,yes,no,no,yes,yes,no,no,no,no,BILIR_low,62.0,166.0,ALBU_high,63
38.0,female,yes,no,no,no,no,yes,no,no,no,no,no,BILIR_low,53.0,42.0,ALBU_high,85.0,
38.0,female,no,yes,no,no,no,no,yes,no,no,no,no,BILIR_low,70.0,28.0,ALBU_high,62.0,
```

Fig. 15. A part of the transformed “hepatitis.arff” dataset

In this case, the output of NIP is the following: 1) the file hepatitis-m.arff that contains the low quality dataset, 2) the specification file hepatitis-m.arff.spec and 3) a file with the fuzzy partition of the attributes.

6. Conclusions

The lack of datasets representing the imperfection inherent in any data collection makes it difficult to develop data mining techniques to manage low quality data. Moreover, it is possible to improve some datasets generated automatically by introducing expert knowledge. With the main objective to improve the creation and management of low quality datasets, in this paper we have presented a tool developed in Java, which we have called NIP. The tool can transform a dataset by adding low quality data, removing and replacing any data, constructing a fuzzy partition of the attributes, etc. And, it also allows different input/output formats of the dataset. This way, we think that the tool can be used as a common framework to create low quality datasets and so, for testing and comparison of data mining techniques and algorithms.

When we compare it with other similar software tools, we can conclude that this tool is able to handle more types of low quality data allowing custom formats in both input and output and so it becomes a more flexible tool.

Acknowledgment

Supported by the project TIN2011-27696-C02-02 of the Ministry of Economy and Competitiveness of Spain. Thanks also to “Fundación Séneca - Agencia de Ciencia y Tecnología de la Región de Murcia” (Spain) for the Funding Program for Research Groups of Excellence (04552/GERM/06) and the support given to Raquel Martínez by the FPI scholarship program.

References

1. J. Han and M. Kamber, *Data mining: Concepts and Techniques*, Morgan Kaufmann Pub., 3rd Edition, 2011.
2. R.O. Duda, P.E. Hart and D.G. Stork, *Pattern classification*, New York: John Wiley and Sons, 2001.
3. D.J.C. Mackay, *Information theory, inference and learning algorithms*, Cambridge: Cambridge University Press, 2003.
4. I.H. Witten and E. Frank, *Data mining: Practical Machine Learning Tools and Techniques.*, San Francisco: Morgan Kaufmann Pub., 2005.
5. L. Sánchez, I. Couso and J. Casillas, *Genetic learning of fuzzy rules based on low quality data*, Fuzzy Sets and Systems, 160:2524-2552, 2009.
6. J.R. Quinlan, *C4.5: Programs for machine learning*, California: Morgan Kaufmann Pub., 1993.
7. Chen-Chia Chuang, *Extended support vector interval regression networks for interval input-output data*, Information Sciences, 178(3):871-891, 2008.
8. J.M. Cadenas, M.C. Garrido and R. Martínez-España, Software tools: NIP Tool, http://heurimind.inf.um.es/, University of Murcia, 2012.
9. R. Hickey, Noise Modeling and Evaluating Learning from Examples, *Artificial Intelligence*, 82(1–2):157–179, 1996.
10. C.E. Brodley, M.A. Friedl, Identifying mislabeled training data, *J Artif Intell Res*, 11:131167, 1999.
11. P.P Bonissone, Approximate Reasoning Systems: Handling Uncertainty and Imprecision in Information Systems, In A. Motro and Ph. Smets (eds.), *Uncertainty Management in Information Systems: From Needs to Solutions*, pp.369–395, Kluwer Academic Publishers, 1997.
12. R. Coppi, M.A. Gil and H.A.L. Kiers, The fuzzy approach to statistical analysis, *Computational Statistics & Data Analysis*, 51:1–14, 2006.

13. D. Dubois and H. Prade, *Possibility Theory: An approach to Computerized Processing of Uncertainty*, New York: Plenum Press, 1988.
14. D. Dubois and D. Guyonnet, Risk-informed decision-making in the presence of epistemic uncertainty, *International Journal of General Systems*, 40(2):145–167, 2011.
15. Angoss Software Corporation, Knoweledge SEEKER: Data mining software with predictive analytics made intuitive for business and technical users. http://http://www.angoss.com, 2012.
16. J. Little and C. Moler, The Mathworks. Simulink Design Optimization, 1.1 edition, Natic, MA, 2009.
17. E. Diday and M. Noirhomme-Fraiture, *Symbolic Data Analysis and the SODAS Software*, New York: Wiley Interscience, 2008.
18. J. Alcalá-Fdez, L. Sánchez, S. García, M.J. del Sesus, S. Ventura, J.M. Garrell, J. Otero, C. Romero, J. Bacardit, V.M. Rivas, J.C. Fernández and F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13:307–318, 2009.
19. J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez and F. Herrera, KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Multiple-Valued Logic and SoftComputing*, 17(2–3):255–287, 2011.
20. I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz and T. Euler, YALE: Rapid Prototyping for Complex Data Mining Tasks, In *Proc. 12th ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining*, pp. 935–940, 2006.
21. K. Morik and M. Scholz, The MiningMart Approach to Knowledge Discovery in Databases, In eds. N. Zhong, J. Liu (eds.), *Intelligent Tech. for Information Analysis*, pp. 47–65, Berlin: Springer, 2004.
22. J. Rushing, R. Ramachandran, U. Nair, S. Graves, R. Welch and H. Lin, ADaM: a data mining toolkit for scientists and engineers, *Computers & geosciences*, 31(5):607–618, 2005.
23. X. Llorà, E2K: Evolution to knowledge, *SIGEVOlution*, 1(3):10-16, 2006.
24. M.R Berthold, N. Cebron, F. Dill, T.R. Gabriel, T. Ktter, T. Meinl, P. Ohl, C. Sieb, K. Thiel and B.Wiswedel, KNIME: The Konstanz Information Miner, In C. Preisach, H. Burkhardt, L. Schmidt-Thieme and R. Decker (eds.), *Data Analysis, Machine Learning and Applications*, pp. 319-326, Berlin: Springer, 2008.
25. J. Demar, B. Zupan, G. Leban and T. Curk, Orange: From experimental machine learning to interactive data mining, *Lecture Notes in Computer Science*, 3202:537–539, 2004.
26. R. Rakotomalala, TANAGRA: a free software for research and academic purposer, *In Proceedings of EGC'2005, RNTI-E-3*, 2:697-702, 2005.
27. J.M. Cadenas, M.C. Garrido, R. Martínez-España, P.P. Bonissone, OFP_CLASS: A hybrid method to generate Optimized Fuzzy Partitions for Classification, *Soft Computing*, 16(4):667–682, 2012.
28. Y-S. Choi, B.R. Moon, Feature Selection in Genetic Fuzzy Discretization for the Pattern Classification Problems, *IEICE Transac*, 90(7):1047–1054, 2007.
29. X. Zhu, X. Wu, Y. Yang, Error detection and impact-sensitive instance ranking in noisy datasets, *In Proceedings of 19th National conference on Artificial Intelligence (AAAI-2004)*, 378–383, 2004.
30. G. Batista and M. Monard, An Analysis of Four Missing Data Treatment Methods for Supervised Learning, *Applied Artificial Intelligence*, 17(5-6):519-533, 2003.
31. J. Luengo, S. García, F. Herrera, A study on the use of imputation methods for experimentation with Radial Basis Function Network classifiers handling missing attribute values: The good synergy between RBFNs and EventCovering method, *Neural Networks*, 23:406-418, 2010.
32. A. Asuncion, and D.J. Newman, UCI Machine Learning Repository, http://www.ics.uci.edu/∼mlearn /ML-Repository.html, Irvine, University of California, School of Inf. and Computer Science, 2007.
33. A. Palacios, M.J. Gacto, J. Alcalá-Fdez, Mining Fuzzy Association Rules from Low Quality Data. *Soft Computing*, 16(5): 883-901, 2012.