A First Approach in the Class Noise Filtering Approaches for Fuzzy Subgroup Discovery

C. J. Carmona and J. Luengo

Abstract The presence of noise in data is a common problem that produces several negative consequences, and is an unavoidable problem, which affects the data collection and data preparation processes in Data Mining applications, where errors commonly occur. The performance of the models built under such circumstances will heavily depend on the quality of the training data. Hence, problems containing noise are complex problems and accurate solutions are often difficult to achieve without using specialized techniques. A particular supervised learning field as subgroup discovery has overlooked the analysis of noise and its impact in the description obtained. In this paper, the noise impact in subgroup discovery is analyzed in a complete experimental study, using recent filtering techniques for several class noise levels. Specifically, the analysis is performed through the FuGePSD algorithm which is a state-of-the-art SD algorithm based on genetic programming and fuzzy logic.

Keywords Subgroup discovery · Class noise · Noise filters

1 Introduction

Real-world data is never perfect and often suffers from corruptions that may harm interpretations of the data, models built and decisions made. Noise can negatively affect the system performance in terms of accuracy, building time, size and interpretability of the model built [23] and it is specially relevant in supervised problems, where it alters the relationship between the informative features and the measure output. For this reason noise has been specially studied in classification and regression where noise hinders the knowledge extraction from the data and spoils the models

C.J. Carmona (✉) · J. Luengo
Dept. of Civil Engineering, University of Burgos, 09006 Burgos, Spain
e-mail: cjcarmona@ubu.es

J. Luengo
e-mail: jluengo@ubu.es

Á. Herrero et al. (eds.), *10th International Conference on Soft Computing Models in Industrial and Environmental Applications*, Advances in Intelligent Systems and Computing 368, DOI 10.1007/978-3-319-19719-7_34

when compared to models learned from clean data for the same domain, which represent the real implicit knowledge of the problem [10].

Several approaches have been studied in the literature to deal with noisy data and to obtain better models, traditionally in classification problems. *Robust learners* [2] are characterized for being less influenced by noise, but designing a robust learner is not trivial. C4.5 [18] is a typical example thanks to its pruning phase. Several works [19] claim that complete or partial noise correction made by *data polishing methods* improves test performance results in comparison with no preprocessing, but it is only feasible in small data sets due to a high computational cost. Finally, the most popular choice are *noise filters* [3, 13], as they act as a preprocessing step, identifying and eliminating the noisy instances from the training data.

However, the effect of noise and capabilities of descriptive algorithms in the presence of noise has been mostly ignored, and since this framework of data mining also relies on supervised examples, the negative effects of noise cannot be ignored. Subgroup discovery (SD) [12] is a descriptive data mining technique using supervised learning, i.e. it is a half-way between classification and description, where the knowledge is represented through rules. The analysis of quality measures in SD is a key factor in order to observe the correct operation of the SD algorithms. The values of these quality measures can be affected for the presence of noise in the data.

In this work we are interested in analysing the performance of SD learning in the presence of noise, and to study different approaches to deal with it. Since noise filtering is a popular preprocessing step that does not require any modification of the SD algorithms, we will use three recent filters for class noise: EF, CVCF and IPF. From 14 base supervised data sets, different amounts of class noise will be introduced, from 5 to 20 %, generating an increasingly noisy scenario to check the suitability of the filtering for SD. We will use a state-of-the-art SD technique, FuGePSD, which is the most recent evolutionary fuzzy system (EFS) for SD presented up to the moment and able to deal with low amounts of noise. The fuzzy confidence measure is used to evaluate the performance of FuGePSD, and the analysis is supported by the use of the Wilcoxon's Signed Rank non-parametric statistical test.

The rest of this contribution is organized as follows. Section 2 introduces the background concepts of subgroup discovery and filtering techniques used. Next, Sect. 3 describes the experimental framework and includes the experimental results and their analysis. Finally, Sect. 4 presents some concluding remarks.

2 Preliminaries

This section presents main concepts, algorithms and noise filters used in the paper. Section 2.1 presents main properties of SD and the FuGePSD algorithm, Sects. 2.2, 2.3 and 2.4 presents noise filters employed in the experimental study.

2.1 Subgroup Discovery

SD is a descriptive data mining technique based on supervised learning. The concept of SD was initially introduced by Kloesgen [14] and Wrobel [21]. The main purpose of SD is to seek and explore relationships between different properties or variables with respect to a target variable, and representations of the knowledge are performed through rules which consist of induced subgroup descriptions [9, 16]. Each rule R can be formally defined as:

$$R : Cond \rightarrow Target_{value}$$

where $Target_{value}$ is a value for the variable of interest (target variable) for the SD task (which also appears as *Class* in the literature), and *Cond* is commonly a conjunction of features (attribute-value pairs) which is able to describe an unusual statistical distribution with respect to the $Target_{value}$.

Despite the use of a target variable, SD is a descriptive induction task using supervised learning while classification is a predictive task. Main differences between SD and classification can be observed in [12].

The most important elements considered for an SD approach are: the target variable, the search strategy, the descriptive language of the subgroups and the quality measures used. Reviews about major properties, features, algorithms and real-world problems solved through the application of SD algorithms can be found in [4, 12].

Throughout the literature there are a wide number of SD algorithms based on exhaustive or stochastic strategies, among others. Recently, a new algorithm called FuGePSD has been presented [5]. This algorithm is an EFS [11] which are basically a fuzzy system augmented by a learning process based on evolutionary computation [8]. Fuzzy systems are usually considered in the form of fuzzy-rule based systems (FRBSs), which are composed of "IF-THEN" rules where both the antecedent and consequent can contain fuzzy logic statements and EAs are well known and widely used global search techniques with the ability to explore a large search space. In summary, the properties of this type of systems make them highly suitable for the development of SD approaches. In fact, the use of fuzzy rules, based on fuzzy logic [22], already allow to consider uncertainty, and also to represent the continuous variables in a manner which is close to human reasoning. In this way, interpretable fuzzy rules consider continuous variables as linguistic ones, where values are represented through fuzzy linguistic labels (*LLs*). The fuzzy set corresponding to each *LL* can be specified by the user or defined by means of uniform partitions if knowledge is not available.

Equation 1 represents a canonical fuzzy rule:

$$R : IF\ X_1 = (LL_1^2)\ \ AND\ \ X_3 = (LL_3^1)\ THEN\ Target_{value} \tag{1}$$

where:

- $X = \{X_m / m = 1, \ldots, n_v\}$ is a set of features used to describe the subgroups, and n_v is the number of descriptive features.

- $T = \{Target_{value}/j = 1, \ldots, n_{tv}\}$ is a set of values for the target variable, and n_{tv} is the number of values for the target variable.
- $LL_{n_v}^{l_{n_v}}$ is the LL number l_{n_v} of the variable n_v.

FuGePSD is based on genetic programming [15] with the ability to extract descriptive fuzzy rules for the SD task. It employs a tree with a variable-length structure to represent the individuals of the population together several mechanisms and specific operators in order to maximise the quality measures. A complete description of the algorithm FuGePSD can be found in [5].

2.2 Ensemble Filter

The *Ensemble Filter* (EF) [3] uses a set of learning algorithms to create classifiers in several subsets of the training data that serve as noise filters for the training set. The identification of potentially noisy instances is carried out by performing an Γ-FCV on the training data with μ classification algorithms, called filter algorithms. In the developed experimentation for this contribution we have utilized the three filter algorithms used by the authors in [3], which are C4.5, 1-NN and LDA [17]. The complete process carried out by EF is described below:

- Split the training data set D_T into Γ equal sized subsets.
- For each one of the μ filter algorithms:
  - For each of these Γ parts, the filter algorithm is trained on the other $\Gamma - 1$ parts. This results in Γ different classifiers.
  - These Γ resulting classifiers are then used to tag each instance in the excluded part as either correct or mislabeled, by comparing the training label with that assigned by the classifier.
- At the end of the above process, each instance in the training data has been tagged by each filter algorithm.
- Add to D_N the noisy instances identified in D_T using a consensus voting scheme, taking into account the correctness of the labels obtained in the previous step by the μ filter algorithms.
- Remove the noisy instances from the training set: $D_T \leftarrow D_T \setminus D_N$.

2.3 Cross-Validated Committees Filter

The *Cross-Validated Committees Filter* (CVCF) [20] uses ensemble methods in order to preprocess the training set to identify and remove mislabeled instances in classification data sets. CVCF is mainly based on performing an Γ-FCV to split the full training data and on building classifiers using decision trees in each training

subset. The authors of CVCF place special emphasis on using ensembles of decision trees such as C4.5. The basic steps of CVCF are the following:

- Split the training data set D_T into Γ equal sized subsets.
- For each of these Γ parts, C4.5 (as suggested by the authors) is trained on the other $\Gamma - 1$ parts. This results in Γ different classifiers.
- These Γ resulting classifiers are then used to tag each instance in the training set D_T as either correct or mislabeled, by comparing the training label with that assigned by the classifier.
- Add to D_N the noisy instances identified in D_T using a voting scheme (the majority scheme in our experimentation), taking into account the correctness of the labels obtained in the previous step by the Γ classifier built.
- Remove the noisy instances from the training set: $D_T \leftarrow D_T \setminus D_N$.

2.4 Iterative-Partitioning Filter

The *Iterative-Partitioning Filter* (IPF) [13] is a preprocessing technique based on the *Partitioning Filter* [24]. It is employed to identify and eliminate mislabeled instances in large data sets. Most noise filters assume that data sets are relatively small and capable of being learned after only one time, but this is not always true and partitioning procedures may be necessary.

IPF removes noisy instances in multiple iterations until a stopping criterion is reached. The iterative process stops if, for a number of consecutive iterations s, the number of identified noisy instances in each of these iterations is less than a percentage p of the size of the original training data set. Initially, we have a set of noisy instances $D_N = \emptyset$ and a set of good data $D_G = \emptyset$. The basic steps of each iteration are:

- Split the training data set D_T into Γ equal sized subsets.
- For each of these Γ parts, C4.5 is trained on this part as recommended by the authors. This results in Γ different trees.
- These Γ resulting classifiers, are then used to tag each instance in the training set D_T as either correct or mislabeled, by comparing the training label with that assigned by the classifier.
- Add to D_N the noisy instances identified in D_T using majority voting, taking into account the correctness of the labels obtained in the previous step by the Γ classifier built.
- Remove the noisy instances and the good data from the training set: $D_T \leftarrow D_T \setminus \{D_N \cup D_G\}$.

At the end of the iterative process, the filtered data is formed by the remaining instances of D_T and the good data of D_G; that is, $D_T \cup D_G$.

3 Experimental Analysis

First, this section describes the processes to induce class noise into original base data sets and the methodology for the analysis of the results (Sect. 3.1). Next, the results and their analysis are presented in Sect. 3.2.

3.1 Experimental Framework and Analysis Methodology

The experimentation is based on 14 real-world classification problems from the KEEL-data set repository[1] [1] shown in Table 1, with their characteristics. Their initial amount of class noise is unknown and thus no assumptions about the noise level can be made. In order to control the amount of noise in each data set, different class noise levels *x%* are introduced with the *uniform class noise* scheme [19]. Following this noise introduction procedure, *x%* of the examples are corrupted by randomly replacing its current class label for any other possible one, drawn from a discrete uniform distribution.

Table 1 Base data sets used to introduce noise, including the number of attributes and their type (real, integer or nominal), the number of examples and the number of classes for each one

Data set	# Attributes (R/I/N)	# Examples	# Classes
Balance	4 (4/0/0)	625	3
Banana	2 (2/0/0)	5300	2
Cleveland	13 (13/0/0)	303	5
Ecoli	7 (7/0/0)	336	8
German	20 (0/7/13)	1000	2
Heart	13 (1/12/0)	270	2
Ionosphere	33 (32/1/0)	351	2
Iris	4 (4/0/0)	150	3
Led7digit	7 (7/0/0)	500	10
Newthyroid	5 (4/1/0)	215	3
Phoneme	5 (5/0/0)	5404	2
Pima	8 (8/0/0)	768	2
Vehicle	18 (0/18/0)	846	4
Wine	13 (13/0/0)	178	3

The accuracy estimation of the classifiers in a data set is obtained by means of 3 runs of a stratified 5-fold cross-validation. 5 partitions are used because, if each partition has a large number of examples, the noise effects will be more notable,

[1] http://www.keel.es/datasets.php.

facilitating their analysis. New class noise data sets will be created from the aforementioned forty base data sets, considering the noise levels ranging from $x = 0\,\%$ (base data sets) to $x = 20\,\%$, by increments of $5\,\%$.

In order to check the behavior of the different methods when dealing with class noise, the results of the SD algorithm are evaluated by using a modified quality measure of confidence, the fuzzy confidence [6]:

$$FCnf(R_i) = \frac{\sum_{E^k \in E / E^k \in TargetValue_k} APC(E^k, R_i)}{\sum_{E^k \in E} APC(E^k, R_i)} \tag{2}$$

where, *APC* (Antecedent Part Compatibility) is the degree of compatibility between an example and the antecedent component of a fuzzy rule, i.e., the degree of membership for the example to the fuzzy subspace delimited by the antecedent part of the rule. An example E^k verifies the *APC* of a rule if

$$APC(E^k, R_i) = T(\mu_{LL_1^1}(e_1^k), \ldots, \mu_{LL_{n_v}^{l_{n_v}}}(e_{n_v}^k)) > 0 \tag{3}$$

The FCnf results' analysis are supported by a Wilcoxon's statistical test [7]. This is a non-parametric pairwise test that aims to detect significant differences between two sample medians. For each level of class noise, the three filters (CVCF, EF and IPF) and the no filtering approach will be compared using Wilcoxon's test.

3.2 Noise Filtering for Subgroup Discovery

We want to analyze whether using a filtering approach is beneficial for the performance of FuGePSD, as a recent and representative SD algorithm. For this reason, we compare the capabilities of FuGePSD without filtering, as being a fuzzy approach to SD should make it more robust against noise, over the increasingly noisy versions of the base data sets. We call this absence of filtering "No filter" in Table 2, where the results of FuGePSD after preprocessing the noisy data sets with CVCF, EF and IPF are also shown. We also indicate the base case, in which no class noise is introduced in the data set, in order to study whether the introduction of noise causes a significant decrement in FCnf as the amount of noise increases.

As can be seen from Table 2, there is a change of which approach is the best as the amount of noise increases. While for $5\,\%$ EF seems to be the best approach, being the best for 7 data sets, IPF and CVCF become better options than EF as the noise increases. It is also worthy to note that no filtering can be a reasonable choice for lower amounts of noise, as FuGePSD is able to deal with these small amounts of noise thanks to its fuzzy nature. However, as the noise is too much to be ignored, filtering is mandatory.

We must mention that for some data sets, introducing noise improves the FCnf value when no filtering, due to a beneficial change in the class labels, but it is not the

Table 2 Fuzzy confidence FCnf results for class noise for FuGePSD

	0 %	5 %				10 %				15 %				20 %			
	All	No filter	CVCF	EF	IPF	No filter	CVCF	EF	IPF	No filter	CVCF	EF	IPF	No filter	CVCF	EF	IPF
Balance	0.62	**0.68**	0.49	0.53	0.47	**0.69**	0.45	0.49	0.49	**0.63**	0.46	0.44	0.45	**0.64**	0.44	0.43	0.41
Banana	0.59	**0.59**	0.58	0.57	0.58	**0.59**	0.58	0.57	0.58	**0.58**	0.57	0.56	0.57	**0.57**	**0.57**	**0.57**	**0.57**
Cleveland	0.54	0.52	**0.53**	0.50	0.52	**0.51**	0.49	0.46	0.46	0.45	0.48	0.46	**0.49**	0.46	**0.52**	0.48	0.51
Ecoli	0.50	**0.46**	0.10	0.00	0.11	0.44	**0.51**	0.00	0.11	0.43	**0.52**	0.00	0.00	0.42	**0.49**	0.00	0.29
German	0.73	0.71	0.74	**0.75**	0.74	0.69	**0.74**	0.73	0.73	0.69	**0.72**	**0.72**	**0.72**	0.68	0.69	**0.70**	**0.70**
Heart	0.71	0.70	0.75	0.73	**0.76**	0.69	0.70	0.70	**0.72**	0.68	0.69	0.69	**0.71**	0.69	**0.76**	0.74	0.75
Ionosphere	0.62	0.64	0.90	**0.91**	**0.91**	0.46	0.90	0.90	**0.91**	0.50	0.88	0.87	**0.89**	0.60	0.83	0.83	**0.84**
Iris	0.76	0.74	**0.83**	**0.83**	**0.83**	0.67	**0.85**	**0.85**	**0.85**	0.66	0.79	**0.80**	0.79	0.68	0.74	0.74	**0.75**
Led7digit	0.47	0.46	0.60	**0.62**	**0.62**	0.44	**0.58**	**0.58**	**0.58**	0.40	0.54	0.57	**0.58**	0.40	0.54	0.54	**0.57**
Newthyroid	0.67	0.64	**0.84**	**0.84**	**0.84**	0.66	**0.84**	**0.84**	**0.84**	0.61	0.78	**0.79**	0.78	0.62	**0.86**	**0.86**	**0.86**
Phoneme	0.80	**0.79**	0.74	0.73	0.74	**0.75**	0.71	0.69	0.71	**0.76**	0.70	0.70	0.71	**0.75**	0.70	0.69	0.70
Pima	0.76	0.74	**0.78**	**0.78**	**0.78**	0.72	**0.77**	0.76	0.75	0.73	**0.74**	**0.74**	**0.74**	0.70	**0.75**	0.74	0.73
Vehicle	0.51	0.50	**0.64**	**0.64**	0.63	0.49	0.63	0.62	**0.64**	0.47	**0.60**	0.58	0.59	0.46	**0.58**	0.57	0.56
Wine	0.86	0.80	0.90	**0.92**	0.91	0.77	0.90	**0.92**	0.90	0.72	0.81	**0.85**	0.81	0.73	0.80	**0.81**	0.80

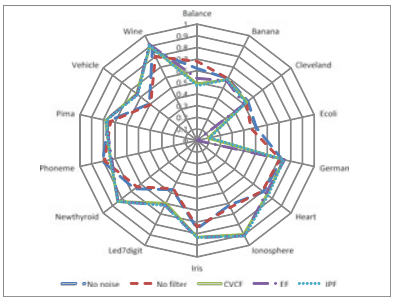

Fig. 1 FCnf of FuGePSD for 5 % class noise

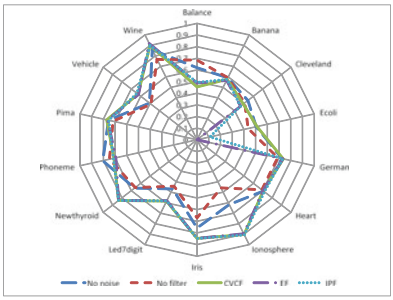

Fig. 2 FCnf of FuGePSD for 10 % class noise

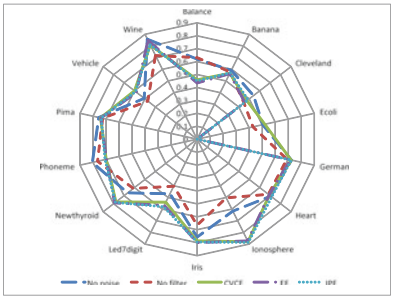

Fig. 3 FCnf of FuGePSD for 15 % class noise

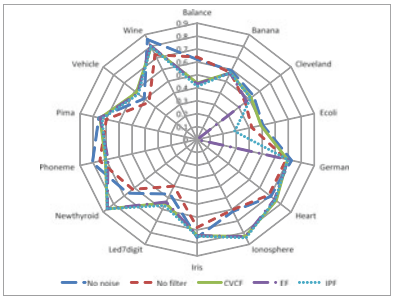

Fig. 4 FCnf of FuGePSD for 20 % class noise

Table 3 Summary of the Wilcoxon test for **class noise** of FuGePSD. • = the method in the row improves the method of the column. ∘ = the method in the column improves the method of the row. Upper diagonal of level significance $\alpha = 0.9$, Lower diagonal level of significance $\alpha = 0.95$

	5 %					10 %					15 %					20 %				
	No noise	No filter	CVCF	EF	IPF	No noise	No filter	CVCF	EF	IPF	No noise	No filter	CVCF	EF	IPF	No noise	No filter	CVCF	EF	
No noise	–	•				–	•				–	•				–	•			
No filter		–				∘	–	∘			∘	–	∘			∘	–	∘		
CVCF			–					–					–				•	–	•	
EF				–					–	∘				–	∘			∘	–	
IPF					–					–					–					–

common case as it only happens for the *balance*, *ecoli* and *phoneme* data sets. It is also important to note that the behavior of the filtering approaches vary depending on the data set. For better visualize this variability, we include the representation of the FCnf values in a radial depiction for each noise level. Figures 1, 2, 3 and 4 show the FCnf values for 5 10, 15 and 20 % of class noise respectively. From them it is easy to observe how *No filter* is similar to *No noise* with 5 % of noise, but becomes a bad option from 10 % onwards. We can also point out how some data sets benefit more from filtering, as *ionosphere*, *german*, *newthyroid*, *iris* and *heart*. For these data sets, the filtering even causes FuGePSD to work better than the *No noise* base case, as the filtering is able to clean the class boundaries and allowing FuGePSD to obtain a better description of the class clusters. For other data sets, as *balance* or *phoneme*, no filtering is appropriate for FuGePSD, causing a decrement in FCnf, but these are scarce cases compared to the total of data sets.

In Table 3 we include the summarized results of the Wilcoxon's Signed Rank test, in which we compare the different approaches for each noise level. Significative differences only appear with higher levels of noise. However, at all noise levels no filtering is worse than the 0 % noise level (*No noise*) but the filtering approaches do not show this difference. Thus, no filtering is significatively worse than the original, noise-induced free, case even at low amounts of noise. As a consequence, filtering in noisy frameworks for SD should be always taken into account. When the amount of noise is non-ignorable (from 10 % onwards), some filtering approaches are better suited for FuGePSD. As can be seen from Table 3, CVCF is specially indicated when the level of noise is high (15 and 20 %). As CVCF is never worse than the rest of filters, and it outperforms no filtering from 10 % onwards, it seems to be the best option for FuGePSD in this contribution and a promising choice for SD approaches.

4 Concluding Remarks

In this contribution, we have analyzed the influence of the class noise in the SD problem, that is a descriptive data mining technique using supervised learning. Specifically, the analysis has been performed with the FuGePSD algorithm which is the most recent EFS presented in the literature. It has an excellent behavior with data sets with continuous variables.

To do so, different class noise amounts (5, 10, 15 and 20 %) have been introduced in the original data sets. The study shows different situations with respect to the increment of noise. As FuGePSD is a fuzzy SD approach, is robust to uncertainty and noise to a certain extent, and thus no filtering can be a reasonable choice for lower amounts of noise. On the other hand, when the amount of noise increases, its treatment cannot be ignored and the application of specialized techniques as noise filters is beneficial. In the latter case, we remark that IPF and CVCF become better options when noise cannot be ignored. In addition, statistical tests support that CVCF seems to be the best option in this contribution and a promising choice for SD approaches.

This work opens future efforts in order to enlarge and observe the influence of the class and/or attribute noise in several SD proposals and so to obtain more comprehensible and complete analysis in this data mining field.

Acknowledgments Supported by the the Spanish Ministry of Economy and Competitiveness under projects TIN2012-33856 (FEDER Founds), the Spanish Ministry of Science and Technology under Projects TIN2011-28488 and TIN2010-15055, and also by the Regional Projects P10-TIC-6858 and P12-TIC-2958.

References

1. Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing 17(2–3):255–287
2. Bonissone P, Cadenas JM, Carmen M (2010) Garrido, and R. Andrés Díaz-Valladares. A fuzzy random forest. International Journal of Approximate Reasoning 51(7):729–747
3. Brodley CE, Friedl MA (1999) Identifying Mislabeled Training Data. Journal of Artificial Intelligence Research 11:131–167
4. Carmona CJ, González P, del Jesus M, Herrera F (2014) Overview on evolutionary subgroup discovery: analysis of the suitability and potential of the search performed by evolutionary algorithms. WIREs Data Mining and Knowledge Discovery 4(2):87–103
5. Carmona CJ, Ruiz-Rodado V, del Jesus M, Weber A, Grootveld M, González P, Elizondo D (2015) A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans. Information Sciences 298:180–197
6. del Jesus MJ, González P, Herrera F, Mesonero M (2007) Evolutionary Fuzzy Rule Induction Process for Subgroup Discovery: A case study in marketing. IEEE Transactions on Fuzzy Systems 15(4):578–592
7. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7:1–30
8. A. E. Eiben and J. E. Smith. Introduction to evolutionary computation. Springer, 2003
9. Gamberger D, Lavrac N (2002) Expert-Guided Subgroup Discovery: Methodology and Application. Journal Artificial Intelligence Research 17:501–527
10. García S, Luengo J, Herrera F (2015) Data Preprocessing in Data Mining. Springer Publishing Company, Incorporated
11. Herrera F (2008) Genetic fuzzy systems: taxomony, current research trends and prospects. Evolutionary Intelligence 1:27–46
12. Herrera F, Carmona CJ, González P, del Jesus MJ (2011) An overview on Subgroup Discovery: Foundations and Applications. Knowledge and Information Systems 29(3):495–525
13. Khoshgoftaar TM, Rebours P (2007) Improving software quality prediction by noise filtering techniques. Journal of Computer Science and Technology 22:387–396
14. W. Kloesgen. Explora: A Multipattern and Multistrategy Discovery Assistant. In Advances in Knowledge Discovery and Data Mining, pages 249–271. American Association for Artificial Intelligence, 1996
15. J. R. Koza. Genetic Programming: On the Programming of computers by Means of Natural Selection. MIT Press, 1992
16. Lavrac N, Cestnik B, Gamberger D, Flach PA (2004) Decision Support Through Subgroup Discovery: Three Case Studies and the Lessons Learned. Machine Learning 57(1–2):115–143
17. G. J. Mclachlan. Discriminant Analysis and Statistical Pattern Recognition (Wiley Series in Probability and Statistics). Wiley-Interscience, 2004

18. J. R. Quinlan. C4.5: programs for machine learning. Morgan Kaufmann Publishers, San Francisco, CA, USA, 1993
19. C.-M. Teng. Correcting Noisy Data. In Proceedings of the Sixteenth International Conference on Machine Learning, pages 239–248, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers
20. S. Verbaeten and A. V. Assche. Ensemble methods for noise elimination in classification problems. In Fourth International Workshop on Multiple Classifier Systems, pages 317–325. Springer, 2003
21. S. Wrobel. An Algorithm for Multi-relational Discovery of Subgroups. In Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery, volume 1263 of LNAI, pages 78–87. Springer, 1997
22. L. A. Zadeh. The concept of a linguistic variable and its applications to approximate reasoning. Parts I, II, III. Information Science, 8–9:199–249,301–357,43–80, 1975
23. Zhu X, Wu X (2004) Class Noise vs. Attribute Noise: A Quantitative Study. Artificial Intelligence Review 22:177–210
24. X. Zhu, X. Wu, and Q. Chen. Eliminating class noise in large datasets. In Proceeding of the Twentieth International Conference on Machine Learning, pages 920–927, 2003