Adv Data Anal Classif (2017) 11:711–730
https://doi.org/10.1007/s11634-016-0260-z

REGULAR ARTICLE

Fuzzy rule based classification systems for big data with MapReduce: granularity analysis

Alberto Fernández[1] · Sara del Río[1] · Abdullah Bawakid[2] · Francisco Herrera[1,2]

Received: 8 November 2015 / Revised: 24 May 2016 / Accepted: 25 May 2016 /
Published online: 6 June 2016

Abstract Due to the vast amount of information available nowadays, and the advantages related to the processing of this data, the topics of big data and data science have acquired a great importance in the current research. Big data applications are mainly about scalability, which can be achieved via the MapReduce programming model.It is designed to divide the data into several chunks or groups that are processed in parallel, and whose result is "assembled" to provide a single solution. Among different classification paradigms adapted to this new framework, fuzzy rule based classification systems have shown interesting results with a MapReduce approach for big data. It is well known that the performance of these types of systems has a strong dependence on the selection of a good granularity level for the Data Base. However, in the context of MapReduce this parameter is even harder to determine as it can be also related with the number of Maps chosen for the processing stage. In this paper, we aim at analyzing the interrelation between the number of labels of the fuzzy variables and the scarcity of the data due to the data sampling in MapReduce. Specifically, we consider that as the partitioning of the initial instance set grows, the level of granularity necessary to

✉ Alberto Fernández
alberto@decsai.ugr.es

Sara del Río
srio@decsai.ugr.es

Abdullah Bawakid
abawakid@kau.edu.sa

Francisco Herrera
herrera@decsai.ugr.es

1 Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain

2 Faculty of Computing and Information Technology, King Abdulaziz University (KAU), Jeddah, Saudi Arabia

achieve a good performance also becomes higher. The experimental results, carried out for several Big Data problems, and using the Chi-FRBCS-BigData algorithms, support our claims.

Keywords Big data · Fuzzy rule based classification systems · Granularity · MapReduce · Hadoop

Mathematics Subject Classification 68W10 Parallel algorithms · 68T05 Learning and adaptive systems · 68T10 Pattern recognition · 68T37 Reasoning under uncertainty

1 Introduction

Within the information technology field, the "Big Data" term is growing in importance during the last years for both academia and industry (Fernández et al. 2014). It refers to the problem related to the management of a large *volume* of information, that arrives at a high *velocity* and under a *variety* of formats (Zikopoulos et al. 2011). It is straightforward to acknowledge that standard processing methods are no longer appropriate to scale to this type of data (Madden 2012).

The challenges derived from collecting and processing these vast amounts of data implies some interesting advantages (Chen and Zhang 2014). In particular, the concept of Data Science arises from the application of Data Mining techniques (Witten et al. 2011) onto these types of Big Data problems (Mattmann 2013; Provost and Fawcett 2013a). Allowing a good scalability of Data Mining solutions, the acquisition of a higher and more accurate degree of knowledge from data can be easily achieved (Wu et al. 2014).

The MapReduce model (Dean and Ghemawat 2010) has arisen as a new methodology for the sake of enabling the development of fast and scalable applications for current researchers and data scientists. It consists on a programming model that allows, in a transparent way, to carry out a distributed execution of the program, together with a robust fault tolerance mechanism (Fernández et al. 2014). In a nutshell, the user only has to codify two main key functions, *Map* and *Reduce*, aimed for splitting the data for processing, and collecting and aggregating the results respectively. This way, it is mandatory to adapt and redesign standard learning algorithms considering these requirements (Chen and Zhang 2014; Wu et al. 2014).

This work focuses on the topic of classification for big data problems. The objective of this task is to apply a learning procedure from a set of labeled examples (training data) to obtain a system that models the problem space by means of their input attributes. Then, when unseen examples (test data) arrive the system, an inference mechanism is applied to determine the label to which this query instance belongs.

There are plenty of different techniques for learning classification systems. Among them, rule induction algorithms aim at discovering a description for the target concept in the form of explicit rules formulated in terms of tests for certain values of the attributes (Fernández et al. 2010). They become very popular as they have a close way to humans for the knowledge representation.

In particular, we must stress the success of Fuzzy Rule Based Classification Systems (FRBCSs) (Ishibuchi et al. 2004) in modeling complex problems. This is due to the proper management of the uncertainty achieved by fuzzy sets, as well as their interpretability based on the linguistic variables (Gacto et al. 2011). In the context of Big Data problems, the former properties make them a valuable tool for developing robust solutions, handling the variety and veracity inherent to the available data.

Specifically, in Río et al. (2015) authors proposed the first FRBCS adapted to the MapReduce scheme, named as Chi-FRBCS-BigData. Considering the limitations of standard fuzzy learning approaches for high dimensionality and large number of instances, this novel approach enables a good scalability, in terms of both execution time and accuracy, to millions of examples and half a hundred of attributes.

It is well known that the granularity level has a significant influence on the FRBCS performance (Cordón et al. 2000). The number of fuzzy labels of each partition can be viewed as a sort of context information. Therefore, depending on the problem structure and the data scattering, a higher or lower number of linguistic terms will be needed in order to provide a good discrimination ability.

The main contributions of this research paper are summarized next:

1. We aim at studying the dependency of the granularity level, i.e. number of fuzzy labels per variable, and the number of selected Maps for the FRBCS in Big Data problems. In order to do so, we have selected the Chi-FRBCS-BigData algorithm in an experimental framework with datasets of different characteristics (especially in terms of number of examples).
2. Our experimental results determine the benefit of a higher granularity in this context for all case studies. This is due to the use of better local representation of the generated subproblems in addition to the final fusion of rules.
3. Finally, we show that FRBCSs are robust in terms of the data scattering, maintaining a good performance even when increasing the number of Maps used. In this way, this type of systems has the advantage of their scalability.

In order to do so, this paper is structured as follows. First, Sect. 2 provides a brief introduction of Big Data, showing how this problem can be addressed by means of the MapReduce programming model. Next, Sect. 3 introduces some preliminary concepts on FRBCSs, describing the Chi-FRBCS-BigData approach. The core of this study is shown in Sect. 4, in which we first stress the significance of granularity in this particular context, then we describe the experimental framework, and finally we show the experimental results to analyze the relationship between the granularity and the number of Maps selected. To conclude the paper, the main findings are summarized in Sect. 5.

2 Big data and the MapReduce programming model

This section is devoted to introduce some concepts on big data (Sect. 2.1). Then, we present an overview of the MapReduce programming model (Sect. 2.2).

2.1 Introduction to big data

Big Data has emerged as a hot topic in the recent years (Chen and Zhang 2014; Fernández et al. 2014; Kambatla et al. 2014). It refers to those advantages, and also challenges, derived from collecting and processing vast amounts of data (Marx 2013; Kraska 2013). The benefits from the management of these types of problems is clear: the larger the data, the higher the degree of knowledge that can be extracted from it (Wu et al. 2014). However, traditional computational approaches are no longer able to scale to the size of Peta-bytes or Zeta-bytes to obtain results in a "tolerable elapsed time." In addition to the former, the speed rate of incoming information is becoming higher and higher. Finally, the different sources that carry out the data recording also implies an heterogeneous structure.

All these facts are known as the 3Vs model of Big Data (*Volume*, *Velocity* and *Variety*), which was the first definition of the Big Data term. Later, the model was extended to 5Vs with: *Value*, which characterizes the business value and, *Veracity*, which is an indicator of data integrity and the trust on this information in order to make decisions. More recently, additional Vs have been added to the model in order to complete the definition of the term reaching at 16Vs, such as *Viability*, *Validity*, *Variability*, *Volatility*, *Vagueness*, *Visualization*, *Venue*, *Vocabulary*, *Verbosity*, *Vulnerability* and *Verification* (Fernández et al. 2014).

As we have stressed above, there are several conditions that must be taken into account in order to consider a given application to be under the umbrella of Big Data. Specifically, these 'V' properties should be regarded in a given context, as to set a particular *threshold* is not straightforward. For example, when the quantity of information that is being processed should be considered "big" enough? The answer is that this concept varies depending on the technology applied to solve the problem (Madden 2012).

Finally, Big Data is about the insight that we want to extract from information, i.e. the actual value of the data. The key here is the analysis that is made for knowledge and business purposes. This situation has lead to the rise of the topic of Data Science (O'Neil and Schutt 2013; Provost and Fawcett 2013b). It can be defined as the process carried out to analyze and get insights from Big Data problems (Waller and Fawcett 2013). In this framework several fields converge, such as machine learning, predictive analysis, and statistics, among others.

2.2 MapReduce programming model

MapReduce (Dean and Ghemawat 2008, 2010) is a distributed programming model for writing massive, scalable and fault tolerant data applications that was developed for processing large datasets over a cluster of machines. The MapReduce model is based on two primary functions: the *Map* function and the *Reduce* function, which must be designed by users. In general terms, in the first phase the input data is processed by the *Map* function which produces some intermediate results; afterwards, these intermediate results will be fed to a second phase in a *Reduce* function which somehow combines the intermediate results to produce a final output.

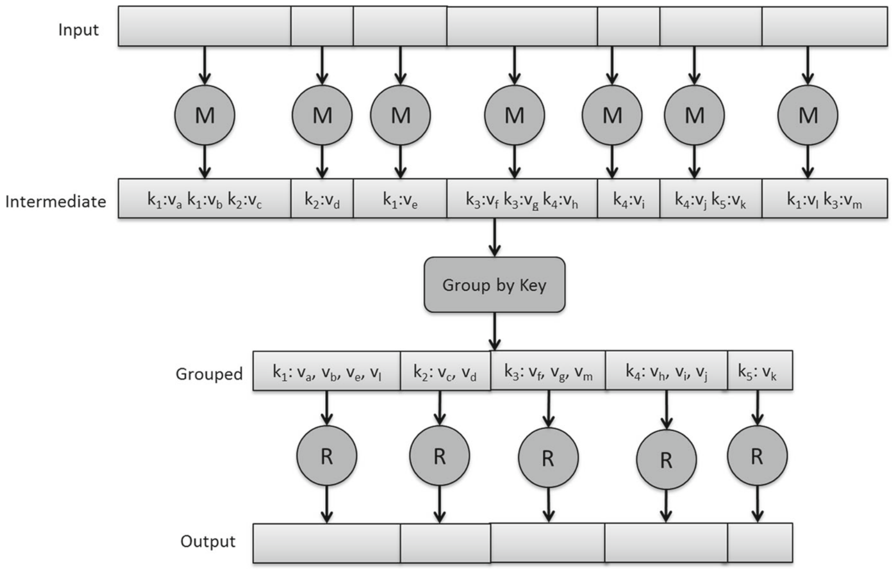

Fig. 1 The MapReduce programming model

The MapReduce model is defined with respect to an essential data structure known as <key,value> pair. The processed data, the intermediate and final results work in terms of <key,value> pairs. In this way, the *map* and *reduce* functions that can be seen in a MapReduce procedure are defined as follows:

- **Map function**: the master node machine takes the input, divides it into several sub-problems and transfers them to the worker nodes. Next, each worker node processes its sub-problem and generates a result that is transmitted back to the master node. In terms of <key,value> pairs, the *Map* function receives a <key,value> pair as input and emits a set of intermediate <key,value> pairs as output. Then, these intermediate <key,value> pairs are automatically shuffled and ordered according to the intermediate key and will be the input to the *Reduce* function.
- **Reduce function**: the master node collects the answers of worker nodes and combines them in some way to form the final output of the method. Again, in terms of <key,value> pairs, the *Reduce* function obtains the intermediate <key,value> pairs produced in the previous phase and generates the corresponding <key,value> pair as the final output of the algorithm.

Figure 1 illustrates a typical MapReduce program with its *Map* and *Reduce* steps. The terms k and v refer to the original key and value pair respectively; k' and v' are the intermediate <key,value> pair that is created after the *Map* step; and v'' is the final result of the algorithm.

Due to the fact that the original MapReduce technology is a proprietary system exploited by Google, different implementations of this framework have made available for public use. Among them, we will focus on the Hadoop MapReduce implementation (Lam 2011) for its wider usage and popularity due to its performance, open source nature, installation facilities and its distributed file system (Hadoop Distributed File System).

3 Fuzzy rule based classification systems for big data

In this section, we first make a short introduction to FRBCS (Sect. 3.1). Then, we describe the novel fuzzy learning algorithm for Big Data used in this work, i.e. the Chi-FRBCS-BigData approach (Sect. 3.2).

3.1 Introduction to fuzzy rule based classification systems

FRBCSs (Ishibuchi et al. 2004) constitute an extension to classical rule-based systems, because they deal with "IF-THEN" rules, but its antecedents and consequents are composed of fuzzy logic statements, instead of classical ones. FRBCSs are composed of a knowledge base (KB), including both the information of the fuzzy sets [(contained in the data base (DB)] and the rules [(within a rule base (RB)], and an inference system.

In this work, the DB will be obtained ad hoc by means of linguistic fuzzy labels (triangular membership functions) homogeneously along the range of each variable. The RB will be extracted using a learning procedure from the input examples, building rules of the following form:

$$\begin{aligned} &\text{Rule } R_j : \text{ If } x_1 \text{ is } A_j^1 \text{ and } \ldots \text{ and } x_n \text{ is } A_j^n \\ &\quad \text{then Class } = C_j \text{ with } RW_j \end{aligned} \tag{1}$$

where R_j is the label of the j-th rule, $\mathbf{x} = (x_1, \ldots, x_n)$ is a n-dimensional pattern vector, A_j^i is an antecedent fuzzy set, C_j is a class label, and RW_j is the rule weight (Ishibuchi and Nakashima 2001). Specifically, we will make use of the heuristic method known as the Penalized Certainty Factor (Ishibuchi and Yamamoto 2005):

$$\text{RW}_j = \text{PCF}_j = \frac{\sum_{x_p \in C_j} \mu_{A_j}(x_p) - \sum_{x_p \notin C_j} \mu_{A_j}(x_p)}{\sum_{p=1}^{m} \mu_{A_j}(x_p)} \tag{2}$$

where $\mu_{A_j}(x_p)$ is the membership degree of x_p, i.e. p-th example of the training set with the antecedents of the rule and C_j is the class determined by rule j.

Finally, regarding the inference system we will apply the fuzzy reasoning method of the single wining rule (Cordón et al. 1999). This methodology predicts as output class that of the rule having the highest matching value overall with the example.

3.2 Chi-FRBCS-BigData algorithm: a MapReduce design

Fuzzy rule learning algorithms are mainly based on building the RB directly from examples. According to this fact, they have a scalability problem for high amounts of data. To address this issue, several approaches have appeared to build parallel fuzzy systems (Hong et al. 2014; Ishibuchi et al. 2013); however, these models aim to reduce the processing time while preserving the accuracy and they are not designed to manage actual Big Data problems.

Recently, the first FRBCS to deal with Big Data was developed in Río et al. (2015), named as Chi-FRBCS-BigData, later extended for imbalanced classification (López et al. 2015), being an area of special interest for Big Data applications (Krawczyk 2016). As its name suggests, this method was based on the Chi et al.'s approach (Chi et al. 1996), which adapts its working procedure to follow a MapReduce scheme. We must point out that the use of the "Chi" algorithm as baseline learning model had the advantage of providing fuzzy rules with same structure, and therefore to be independently created from a subset of examples.

The Chi-FRBCS-BigData approach is based on two different MapReduce processes: the first one is devoted to the building of the fuzzy KB from a Big Data training set; whereas the second procedure is aimed to carry the classification of the examples.

Following the MapReduce paradigm, both schemes distribute the computations along several processing units that manage different chunks of information using Map functions. Then, the obtained results are simply aggregated within the Reduce functions. The whole procedure carried out in Chi-FRBCS-BigData is depicted in Fig. 2, and its stages are described below:

1. *Initial:* the DB is built computing homogeneous fuzzy partitions along the domain of each attribute, depending on the level of granularity selected. Next, the whole training set is divided into independent data blocks which are transferred to the processing units together with the common fuzzy DB.
2. *Map:* In this stage, each processing unit works independently over its available data to build its associated fuzzy RB (called RB_i in Fig. 2) following the original Chi-FRBCS method. Specifically, the procedure iterates among all examples, and each fuzzy rule is created having as antecedent those linguistic terms with the greatest membership degree with the current instance, and its class as consequent. Then, RWs are computed as shown in Equation (2). Finally duplicated rules are removed from the RB and, in case of contradictory consequent, only those with the highest RW are kept.
3. *Reduce:* In this third phase, all RB_i computed by a Map process are aggregated to obtain the final RB (called RB_R in Fig. 2). As rules with the same antecedent may come from different Maps, we follow the **Chi-FRBCS-BigData-Max** scheme in which the most robust rules are maintained in RB_R, i.e. those with the highest weight.
4. *Final:* results computed in the previous phases are provided as the output of the computation process. The generated fuzzy KB is composed by the fuzzy DB built in the "Initial" phase and the fuzzy RB, RB_R, obtained in the "Reduce" phase. This KB will be the model that will be used to predict the class for new examples.

Once we have obtained the whole KB by means of the process that was described above (Fig. 2), we apply an additional and independent MapReduce mechanism to carry out the classification step. This way, the execution of this process can be carried out in a distributed way for the sake of improving the response time of the classifier. Specifically, this class estimation process is depicted in Fig. 3 and consists of the following parts:

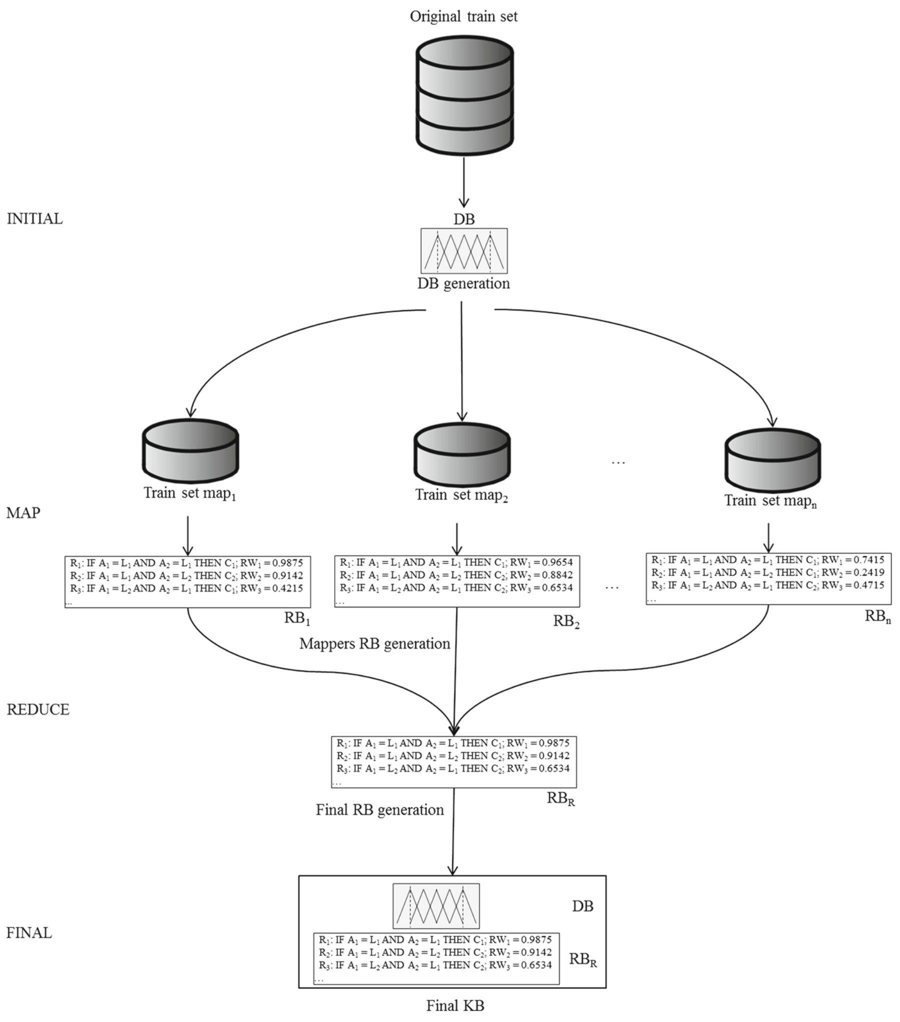
Original train set
INITIAL
DB
DB generation
MAP
Train set map$_1$
Train set map$_2$
...
Train set map$_n$
R$_1$: IF A$_1$ = L$_1$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_1$ = 0.9875
R$_2$: IF A$_1$ = L$_1$ AND A$_2$ = L$_2$ THEN C$_2$; RW$_2$ = 0.9142
R$_3$: IF A$_1$ = L$_2$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_3$ = 0.4215
...
RB$_1$
R$_1$: IF A$_1$ = L$_1$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_1$ = 0.9654
R$_2$: IF A$_1$ = L$_1$ AND A$_2$ = L$_2$ THEN C$_2$; RW$_2$ = 0.8842
R$_3$: IF A$_1$ = L$_2$ AND A$_2$ = L$_1$ THEN C$_2$; RW$_3$ = 0.6534
...
RB$_2$
R$_1$: IF A$_1$ = L$_1$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_1$ = 0.7415
R$_2$: IF A$_1$ = L$_1$ AND A$_2$ = L$_2$ THEN C$_1$; RW$_2$ = 0.2419
R$_3$: IF A$_1$ = L$_2$ AND A$_2$ = L$_1$ THEN C$_2$; RW$_3$ = 0.4715
...
RB$_n$
Mappers RB generation
REDUCE
R$_1$: IF A$_1$ = L$_1$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_1$ = 0.9875
R$_2$: IF A$_1$ = L$_1$ AND A$_2$ = L$_2$ THEN C$_2$; RW$_2$ = 0.9142
R$_3$: IF A$_1$ = L$_2$ AND A$_2$ = L$_1$ THEN C$_2$; RW$_3$ = 0.6534
...
RB$_R$
Final RB generation
FINAL
DB
R$_1$: IF A$_1$ = L$_1$ AND A$_2$ = L$_1$ THEN C$_1$; RW$_1$ = 0.9875
R$_2$: IF A$_1$ = L$_1$ AND A$_2$ = L$_2$ THEN C$_2$; RW$_2$ = 0.9142
R$_3$: IF A$_1$ = L$_2$ AND A$_2$ = L$_1$ THEN C$_2$; RW$_3$ = 0.6534
...
RB$_R$
Final KB

Fig. 2 A flowchart of how the building of the KB is organized in Chi-FRBCS-BigData

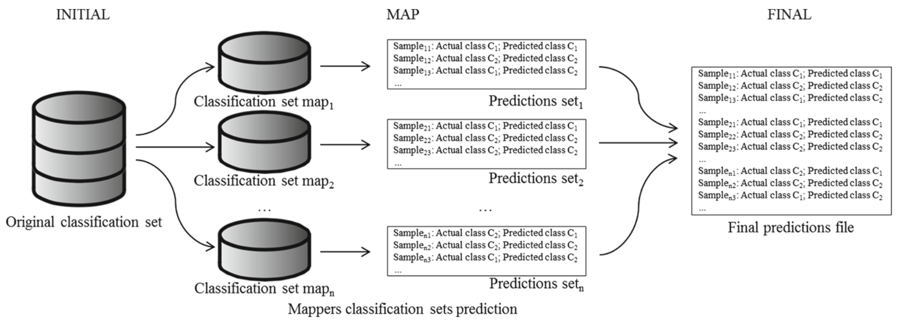
INITIAL
MAP
FINAL
Original classification set
Classification set map$_1$
Classification set map$_2$
...
Classification set map$_n$
Sample$_{11}$: Actual class C$_1$; Predicted class C$_1$
Sample$_{12}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{13}$: Actual class C$_1$; Predicted class C$_2$
...
Predictions set$_1$
Sample$_{21}$: Actual class C$_1$; Predicted class C$_1$
Sample$_{22}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{23}$: Actual class C$_2$; Predicted class C$_2$
...
Predictions set$_2$
...
Sample$_{n1}$: Actual class C$_2$; Predicted class C$_1$
Sample$_{n2}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{n3}$: Actual class C$_1$; Predicted class C$_2$
...
Predictions set$_n$
Mappers classification sets prediction
Sample$_{11}$: Actual class C$_1$; Predicted class C$_1$
Sample$_{12}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{13}$: Actual class C$_1$; Predicted class C$_2$
...
Sample$_{21}$: Actual class C$_1$; Predicted class C$_1$
Sample$_{22}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{23}$: Actual class C$_2$; Predicted class C$_2$
...
Sample$_{n1}$: Actual class C$_2$; Predicted class C$_1$
Sample$_{n2}$: Actual class C$_2$; Predicted class C$_2$
Sample$_{n3}$: Actual class C$_1$; Predicted class C$_2$
...
Final predictions file

Fig. 3 A flowchart of how the classification of a Big Data set is organized in Chi-FRBCS-BigData

1. *Initial:* the original Big Data problem is divided into several chunks so that it can be processed in parallel by the Map functions.
2. *Map:* each map task estimates the class for the examples that are included in its data partition. To do so, each processing unit queries all its associated examples and carries out the fuzzy inference to compute the output class according to the information given in the global KB.
3. *Final:* the class values computed in the previous phase are provided as the output of the computation process. In this case, it is just necessary to concatenate the results provided by each map task.

It is important to point out that this mechanism does not include a "Reduce" step as it is not necessary to perform a computation to combine the results obtained in the "Map" phase.

4 Analysis of the significance of the granularity in a MapReduce approach

In this section, we study the significance of the granularity level for fuzzy rule learning algorithm in Big Data. In particular, we first stress how the number of labels and the number of Maps chosen for the parallel execution are related (subSect. 4.1). Then, we carry out an experimental study which aims at showing the change in behaviour of the Chi-FRBCS-BigData algorithm over different granularity levels and different number of Maps. In order to do so, we first provide some details of the classification problems chosen for the experiments and the configuration parameters (subSect. 4.2). Finally, we present the obtained results that support the hypothesis of this research (subSect. 4.3).

4.1 Significance of the granularity in a MapReduce approach

It is well known that the number of labels for building the DB is a significant parameter to achieve a precise description of the problem space. Particularly, if the classes are highly overlapping, the level of granularity to allow a good discrimination of the examples gets higher. Therefore, the data distribution is one of the main characteristics to determine the most appropriate value of the granularity.

When we obtain the KB within a MapReduce approach, we must be aware that the number of maps selected for this process imply a very different distribution of the data. Hence, as more efficiency is demanded, so it does the scattering of the original data among the map functions.

Regarding this fact, we can assume that the differences in performance for any classification problem regarding the number of Maps selected, will become less significant as the number of labels increases. This is due to several facts. On the one hand, the capabilities of fuzzy learning algorithms make them less sensible to the "lack of data." On the other hand, a higher granularity allows the construction of a robust model, in accordance to the more detailed representation of the space of the problem.

Table 1 Summary of datasets

Datasets	#Ex.	#Atts.	Selected classes	#Samples per class
Kddcup_DOS_vs_normal	4856151	41	(DOS; normal)	(3883370; 972781)
Poker_0_vs_1	946799	10	(0; 1)	(513702; 433097)
Covtype_2_vs_1	495141	54	(2; 1)	(283301; 211840)
Census	141544	41	(−_50000.; 50000+.)	(133430; 8114)
Fars_Fatal_Inj_vs_No_Inj	62123	29	(Fatal_Inj; No_Inj)	(42116; 20007)

Table 2 Configuration parameters for Chi-FRBCS-BigData

Number of labels:	3-5-7-9 fuzzy partitions
Conjunction operator:	Product T-norm
Rule weight:	Penalized certainty factor (Ishibuchi and Yamamoto 2005)
Fuzzy reasoning method:	Winning rule

4.2 Experimental framework

For this study, we have followed the same experimental framework as in the original work in which the Chi-FRBCS-BigData algorithm was proposed (Río et al. 2015). Specifically, five classification problems from the UCI dataset repository (Lichman 2013) have been considered, the KDD Cup 1999 dataset, the Poker Hand dataset, the Covertype dataset, the Census-Income (KDD) dataset and the Fatality Analysis Reporting System (FARS) dataset. A summary of the datasets features is shown in Table 1, where the number of examples (#Ex.), number of attributes (#Atts.), selected classes and the number of examples per class are included. This table is in descending order according to the number of examples of each dataset.

For the experimental analysis, we will take into account the *standard accuracy metric* to evaluate the classification performance. The estimates for this metrics will be obtained by means a 10-fold stratified cross-validation partitioning scheme, i.e., ten random partitions of data with a 10 the combination of nine of them (90 remaining one as test set. The results obtained for each dataset are the average results obtained by computing the mean of all the partitions.

The configuration parameters for the Chi-FRBCS-BigData are presented in Table 2 being "Conjunction operator" the operator used to compute the compatibility degree of the example with the antecedent of the rule and the operator used to compute the compatibility degree and the RW. We must stress that three different levels of granularity have been selected for this study, i.e. low granularity with 3 labels per variable, medium granularity with 5 fuzzy partitions, and high granularity with 7 labels (a short study is also carried out with 9). We must recall that regarding the "Reduce" stage we will make use of the *Chi-FRBCS-BigData-Max* version, as stated in Sect. 3.2.

Additionally, another significant parameter used in our experiments is the number of *Maps* associated to the computation within the MapReduce procedure. We must recall that this value represents the number of subsets of the original dataset that are created. In order to allow the experimental framework to comprise a solid number of case studies, we have selected five different configurations with high scalability, including 32, 64, 128, 256 and 512 maps.

Finally, regarding the infrastructure used to perform the experiments, we have used the research group's cluster with 16 nodes connected with a 40Gb/s Infiniband. Each node is equipped with two Intel E5-2620 microprocessors (at 2 GHz, 15 MB cache) and 64GB of main memory running under Linux CentOS 6.6. The head node of the cluster is equipped with two Intel E5645 microprocessors (at 2.4 GHz, 12 MB cache) and 96GB of main memory. Furthermore, the cluster works with Hadoop 2.6.0 (Cloudera CDH5.4.0), where the head node is configured as name-node and job-tracker, and the rest are data-nodes and task-trackers.

4.3 Analysis of granularity vs. data scattering for MapReduce

In this section we show the experimental results to study the relationship between the granularity and the number of Maps. With this aim, Table 3 summarizes the former information for the five selected problems.

From these performance values, we may extract some interesting information that allows to confirm our hypothesis:

1. The best results overall are achieved with the highest granularity level, i.e. 7 labels per variable. In particular, the loss in performance is so low in average when increasing the number of Maps, that we cannot stress a single configuration as the best one.
2. In all cases the performance per problem increases as it does the number of labels, with the exception of the "Poker" problem between 3 and 5 fuzzy sets per variable.
3. Finally, and as stated in the first point, we may observe a better stability for the data scattering problem (more Maps are used) in those case studies with a higher granularity. With this parameter configuration of Chi-FRBCS-BigData, the relative differences among the results are lower than for those with the coarse-grained case.

With the sake of providing additional support to the previous analysis, we have included in Table 4 a short study with 9 labels using the "fars" dataset. We may observe that the trend shown for Table 3 is also depicted here, for which the higher the granularity, the better the results, independently of the number of Maps used.

In order to complement our experimental study, Fig. 4 depicts the behavior of the model according to the granularity and number of Maps. Specifically, we shown the percentage of loss in accuracy with respect to the baseline case study, i.e. the use of 32 Maps, from which the highest results are usually achieved. All single problems are represented, as well as a summary with the average results as a general approximation of the behavior (Fig. 4f). This way, we may observe graphically our previous highlights, from which the relative differences of the results from 32 Maps to 512 Maps decreases as we apply a higher granularity.

In these figures, the lower the size of the bars, together with smaller variation between the values shown for different case studies (number of Maps) imply a more stable configuration for the granularity parameter. The only exception is the "Kdd-

Table 3 Average results in test for Chi-FRBCS-BigData using 32, 64, 128, 256 and 512 maps

Datasets	32 maps					
	Accuracy			AUC		
	3 Labels	5 Labels	7 Labels	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	99.92498	99.95345	**99.95500**	0.99931	**0.99965**	0.99964
Poker_0_vs_1	58.92973	56.87383	**59.98018**	0.57427	0.56476	**0.59696**
Covtype_2_vs_1	74.61723	83.24885	**87.43990**	0.72395	0.82183	**0.86746**
Census	93.48355	94.68176	**95.54433**	0.62273	0.67379	**0.68272**
Fars_Fatal_Inj_vs_No_Inj	94.25642	99.86419	**99.94522**	0.93803	0.99865	**0.99938**
Average	84.24238	86.92441	**88.57293**	0.77166	0.81173	**0.82923**
Datasets	64 maps					
	Accuracy			AUC		
	3 Labels	5 Labels	7 Labels	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	99.92490	99.95167	**99.95543**	0.99930	0.99960	**0.99965**
Poker_0_vs_1	57.94608	56.78075	**59.94370**	0.56612	0.56384	**0.59664**
Covtype_2_vs_1	74.52195	82.84057	**87.20926**	0.72302	0.81810	**0.86509**
Census	93.30058	94.54349	**95.50047**	0.62459	0.67300	**0.68196**
Fars_Fatal_Inj_vs_No_Inj	93.98393	99.82303	**99.94522**	0.93411	0.99823	**0.99938**
Average	83.93549	86.78790	**88.51082**	0.76943	0.81055	**0.82854**

Table 3 continued

Datasets	128 maps					
	Accuracy			AUC		
	3 Labels	5 Labels	7 Labels	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	99.92525	99.95155	**99.95425**	0.99931	0.99958	**0.99963**
Poker_0_vs_1	56.96169	56.79547	**59.93157**	0.55839	0.56398	**0.59648**
Covtype_2_vs_1	74.01238	82.46134	**86.77500**	0.71797	0.81458	**0.86094**
Census	92.97315	94.38729	**95.45372**	0.62539	0.67160	**0.68019**
Fars_Fatal_Inj_vs_No_Inj	93.81633	99.81723	**99.94522**	0.93178	0.99817	**0.99938**
Average	83.53776	86.68257	**88.41195**	0.76657	0.80958	**0.82732**
Datasets	**256 maps**					
	Accuracy			AUC		
	3 Labels	5 Labels	7 Labels	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	99.94026	99.94953	**99.95127**	0.99940	0.99952	**0.99952**
Poker_0_vs_1	57.89980	56.83982	**59.90099**	0.56698	0.56446	**0.59619**
Covtype_2_vs_1	73.77952	81.88070	**86.45340**	0.71438	0.80846	**0.85789**
Census	92.98441	94.37505	**95.40043**	0.62715	0.67679	**0.68057**
Fars_Fatal_Inj_vs_No_Inj	93.88056	99.85333	**99.94522**	0.93461	0.99854	**0.99938**
Average	83.69691	86.57969	**88.33026**	0.76850	0.80956	**0.82671**

Table 3 continued

Datasets	512 maps					
	Accuracy			AUC		
	3 Labels	5 Labels	7 Labels	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	99.94120	99.95085	**99.95425**	0.99941	0.99960	**0.99964**
Poker_0_vs_1	56.79368	56.79511	**59.92133**	0.55805	0.56395	**0.59636**
Covtype_2_vs_1	73.65237	81.34559	**86.15001**	0.71359	0.80289	**0.85538**
Census	92.66282	94.18538	**95.36667**	0.62340	0.67425	**0.68124**
Fars_Fatal_Inj_vs_No_Inj	93.56599	99.84249	**99.94522**	0.93033	0.99843	**0.99938**
Average	83.32321	86.42388	**88.26750**	0.76496	0.80782	**0.82640**

Best result per problem and number of Maps is stressed in boldface. Best overall result per problem is underlined

Table 4 Average results in test for Chi-FRBCS-BigData using 32, 64, 128, 256 and 512 maps on the Fars_Fatal_Inj_vs_No_Inj dataset

	Accuracy				AUC			
	3 Labels	5 Labels	7 Labels	9 Labels	3 Labels	5 Labels	7 Labels	9 Labels
32 maps	94.25642	99.86419	99.94522	**100.00000**	0.93803	0.99865	0.99938	**1.00000**
	3 Labels	5 Labels	7 Labels	9 Labels	3 Labels	5 Labels	7 Labels	9 Labels
64 maps	93.98393	99.82303	99.94522	**100.00000**	0.93411	0.99823	0.99938	**1.00000**
	3 Labels	5 Labels	7 Labels	9 Labels	3 Labels	5 Labels	7 Labels	9 Labels
128 maps	93.81633	99.81723	99.94522	**100.00000**	0.93178	0.99817	0.99938	**1.00000**
	3 Labels	5 Labels	7 Labels	9 Labels	3 Labels	5 Labels	7 Labels	9 Labels
256 maps	93.88056	99.85333	99.94522	**100.00000**	0.93461	0.99854	0.99938	**1.00000**
	3 Labels	5 Labels	7 Labels	9 Labels	3 Labels	5 Labels	7 Labels	9 Labels
512 maps	93.56599	99.84249	99.94522	**100.00000**	0.93033	0.99843	0.99938	**1.00000**

Best result per number of Maps is stressed in boldface. Best overall result is underlined

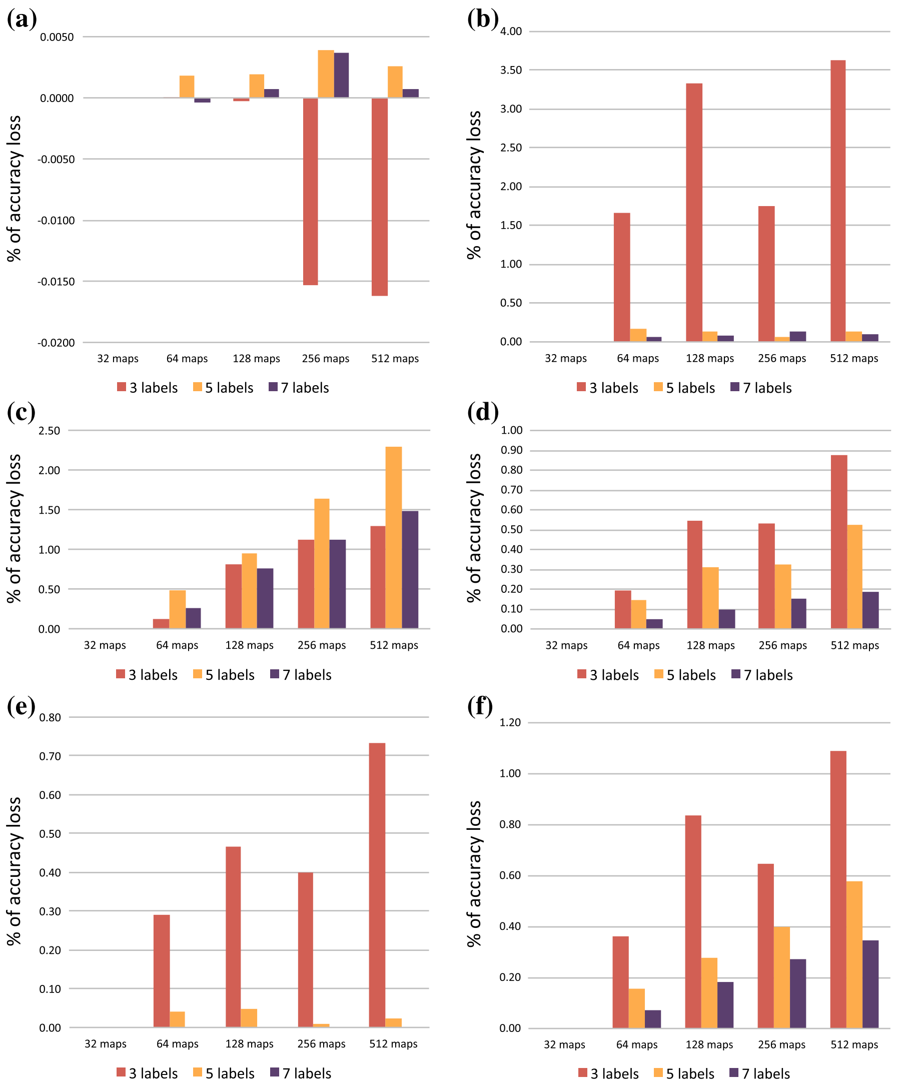

Fig. 4 Percentage of accuracy loss in test for Chi-FRBCS-BigData using 32 to 512 maps. **a** Kddcup_DOS_vs_normal dataset, **b** Poker_0_vs_1 dataset, **c** Covtype_2_vs_1 dataset, **d** census dataset, **e** Fars_Fatal_Inj_vs_No_Inj dataset and **f** average datasets

cup'99" problem, which is a simple classification problem that can be easily addressed using just 3 labels.

Finally, Table 5 shows the elapsed runtime for the training stage of Chi-FRBCS-BigData. From these values, we may extract two interesting conclusions: (1) The lower the granularity, the shorter the required time to obtain the model. (2) The higher the number of Maps, the better the efficiency but with a certain threshold for improvement.

Regarding the first issue, even by applying a rule generation from examples, the search space increases significantly when adding more labels. It was also expected that

Table 5 Average runtime elapsed in seconds for Chi-FRBCS-BigData using 32, 64, 128, 256 and 512 maps

Datasets	32 maps		
	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	**7890.87**	18710.18	25638.19
Poker_0_vs_1	**2210.13**	8675.26	8286.67
Covtype_2_vs_1	**391.40**	4663.55	6891.16
Census	**388.64**	650.44	658.55
Fars_Fatal_Inj_vs_No_Inj	**141.92**	180.77	182.27
Average	**2204.59**	6576.04	8331.37
	64 maps		
	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	**2079.93**	10419.47	15169.54
Poker_0_vs_1	**1635.98**	4592.53	4414.00
Covtype_2_vs_1	**252.19**	3304.88	4282.94
Census	**325.24**	406.94	406.79
Fars_Fatal_Inj_vs_No_Inj	**136.24**	144.19	159.44
Average	**885.91**	3773.60	4886.54
	128 maps		
	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	<u>**1669.02**</u>	11229.52	16564.20
Poker_0_vs_1	<u>**1022.08**</u>	4476.45	4347.99
Covtype_2_vs_1	<u>**189.24**</u>	3623.24	4691.51
Census	<u>**208.05**</u>	457.57	460.93
Fars_Fatal_Inj_vs_No_Inj	<u>**92.74**</u>	178.59	177.62
Average	<u>**636.23**</u>	3993.07	5248.45
	256 maps		
	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	**7774.46**	13681.28	19811.14
Poker_0_vs_1	**4378.83**	4534.92	4382.87
Covtype_2_vs_1	**2883.60**	4158.84	5057.87
Census	**542.56**	568.66	568.10
Fars_Fatal_Inj_vs_No_Inj	**241.31**	245.57	242.83
Average	**3164.15**	4637.85	6012.56
	512 maps		
	3 Labels	5 Labels	7 Labels
Kddcup_DOS_vs_normal	**9602.99**	15784.56	22540.50
Poker_0_vs_1	4492.45	4475.27	**4387.05**
Covtype_2_vs_1	**3880.21**	4910.73	5571.43

Table 5 continued

	512 maps		
	3 Labels	5 Labels	7 Labels
Census	**714.42**	762.38	760.92
Fars_Fatal_Inj_vs_No_Inj	377.25	376.45	**376.11**
Average	**3813.46**	5261.88	6727.20

Fastest configuration per problem and number of Maps is stressed in boldface. Quickest configuration per problem is underlined

this problem will be more accentuated for those datasets with a higher dimensionality, i.e. “KddCup’99”, “CovType” and “Census.”

Focusing on the second point, the limit in the runtime improvement when increasing the number of Maps is mainly due to the high disk overhead in a Hadoop environment. The storing and loading of the data subsets, as well as the time consumption of the communication between processes cause this fact.

In summary, we have stressed an enhancement of the discrimination ability for Chi-FRBCS-BigData by using a higher granularity. This issue may be explain according to two related issues:

- A finer representation of the space problem and therefore a better recognition in the borderline areas, even with fewer examples to learn with.
- The rule fusion in this case results in a more diverse RB. Specifically, we consider that the lower the granularity level, the larger amount of rules are repeated for each independent RB_i. This implies that increasing the number of labels allows rules to be ascribed locally to their own data space, i.e. within their own data chunk, resulting on a better global coverage after the Reduce stage.

5 Concluding remarks

In this paper, we have studied the relationship between granularity and data scattering for FRBCS in the context of Big Data problems. Specifically, we have made use of the Chi-FRBCS-BigData algorithm to accomplish the former analysis.

In the former approach, increasing the value for the number of selected Maps implies a greater division of the original data among the processing units. Our assumption as that the granularity level needed to achieve good results is directly proportional to the number of Maps. In other words, if we aim for efficiency by setting a high number of Map processes, the number of labels to represent the problem shall be also high to avoid a performance decrease.

Throughout a experimental study, carried out with several benchmark problems, we have contrasted our basis for discussion. We have shown that for those problems in which a high granularity level is selected, the accuracy obtained is stable disregard the number of Maps.

Finally, as future work we plan to extend the working procedure of the Chi-FRBCS-BigData algorithm in order to improve its performance and interpretability. First, to allow different granularity levels during the search of the rules in order to enable a better contextualization to the problem. Also to take advantage of the features of the MapReduce programming model in order to extend the Reduce function for decreasing the running time of the algorithm. We can focus on the locality issue of the rules by analyzing their generality adding a new "layer" of Map processes for validation purposes, or even to merge rules by using a double-consequent structure (Cordón and Herrera 2000). An additional possibility consists of the combination of the RBs into an ensemble classifier (Wozniak et al. 2014). This way, we can also take advantage of a wider space partitioning carried out in the Map stage (Jackowski et al. 2014; Wozniak and Krawczyk 2012).

References

Chen CP, Zhang C-Y (2014) Data-intensive applications, challenges, techniques and technologies: a survey on big data. Inf Sci 275:314–347

Chi Z, Yan H, Pham T (1996) Fuzzy algorithms with applications to image processing and pattern recognition. World Scientific, Singapore

Cordón O, Herrera F (2000) A proposal for improving the accuracy of linguistic modeling. IEEE Trans Fuzzy Syst 8(3):335–344

Cordón O, del Jesus M, Herrera F (1999) A proposal on reasoning methods in fuzzy rule-based classification systems. Int J Approx Reason 20(1):21–45

Cordón O, Herrera F, Villar P (2000) Analysis and guidelines to obtain a good fuzzy partition granularity for fuzzy rule-based systems using simulated annealing. Int J Approx Reason 25(3):187–215

Dean J, Ghemawat S (2008) MapReduce: simplified data processing on large clusters. Commun ACM 51(1):107–113

Dean J, Ghemawat S (2010) MapReduce: a flexible data processing tool. Commun ACM 53(1):72–77

Fernández A, Río S, López V, Bawakid A, del Jesus M, Benítez J, Herrera F (2014) Big data with cloud computing: an insight on the computing environment, MapReduce and programming framework. WIREs Data Min Knowl Discov 4(5):380–409

Fernández A, Garcfa S, Luengo J, Bernadó-Mansilla E, Herrera F (2010) Genetics-based machine learning for rule induction: state of the art, taxonomy, and comparative study. IEEE Trans Evolut Comput 14(6):913–941

Gacto MJ, Alcalá R, Herrera F (2011) Interpretability of linguistic fuzzy rule-based systems: an overview of interpretability measures. Inf Sci 181(20):4340–4360

Hong T-P, Lee Y-C, Wu M-T (2014) An effective parallel approach for genetic-fuzzy data mining. Expert Syst Appl 41(2):655–662

Ishibuchi H, Mihara S, Nojima Y (2013) Parallel distributed hybrid fuzzy gbml models with rule set migration and training data rotation. IEEE Trans Fuzzy Syst 21(2):355–368

Ishibuchi H, Nakashima T (2001) Effect of rule weights in fuzzy rule-based classification systems. IEEE Trans Fuzzy Syst 9(4):506–515

Ishibuchi H, Nakashima T, Nii M (2004) Classification and modeling with linguistic information granules: advanced approaches to linguistic data mining. Springer, Berlin

Ishibuchi H, Yamamoto T (2005) Rule weight specification in fuzzy rule-based classification systems. IEEE Trans Fuzzy Syst 13:428–435

Jackowski K, Krawczyk B, Wozniak M (2014) Improved adaptive splitting and selection: the hybrid training method of a classifier based on a feature space partitioning. Int J Neural Syst 24(3):1430007

Kambatla K, Kollias G, Kumar V, Grama A (2014) Trends in big data analytics. J Parallel Distrib Comput 74(7):2561–2573

Kraska T (2013) Finding the needle in the big data systems haystack. IEEE Internet Comput Mag 17(1):84–86

Krawczyk B (2016) Learning from imbalanced data: open challenges and future directions. Progress in Artificial Intelligence, pp 1–12. doi:10.1007/s13748-016-0094-0 (in press)

Lam C (2011) Hadoop in action, 1st edn. Manning, Shelter Island

Lichman M (2013) UCI machine learning repository; university of california, irvine, school of information and computer sciences. http://archive.ics.uci.edu/ml

López V, del Río S, Benítez JM, Herrera F (2015) Cost-sensitive linguistic fuzzy rule based classification systems under the mapreduce framework for imbalanced big data. Fuzzy Sets Syst 258:5–38

Madden S (2012) From databases to big data. IEEE Internet Comput Mag 16(3):4–6

Marx V (2013) The big challenges of big data. Nature 498(7453):255–260

Mattmann CA (2013) Computing: a vision for data science. Nature 493:473–475

O'Neil C, Schutt R (2013) Doing data science, 1st edn. O'Reilly Media, Sebastopol

Provost F, Fawcett T (2013a) Data science and its relationship to big data and data-driven decision making. Big Data 1(1):51–59

Provost F, Fawcett S (2013b) Data science for business. What you need to know about data mining and data-analytic thinking, 1st edn. O'Reilly Media, Sebastopol

Río S, López V, Benítez J, Herrera F (2015) A MapReduce approach to address big data classification problems based on the fusion of linguistic fuzzy rules. Int J Comput Intell Syst 8(3):422–437

Waller M, Fawcett S (2013) Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management. J Bus Logist 34:77–84

Witten IH, Frank E, Hall MA (2011) Data mining: practical machine learning tools and techniques. Morgan Kaufmann series in data management systems. Morgan Kaufmann, Burlington

Wozniak M, Graña M, Corchado E (2014) A survey of multiple classifier systems as hybrid systems. Inf Fusion 16:3–17

Wozniak M, Krawczyk B (2012) Combined classifier based on feature space partitioning. Appl Math Comput Sci 22(4):855–866

Wu X, Zhu X, Wu G-Q, Ding W (2014) Data mining with big data. IEEE Trans Knowl Data Eng 26(1):97–107

Zikopoulos PC, Eaton C, deRoos D, Deutsch T, Lapis G (2011) Understanding big data-analytics for enterprise class hadoop and streaming data, 1st edn. McGraw-Hill Osborne Media, East Windsor