Summarizer: Fuzzy Rule-Based Classification Systems for Vertical and Horizontal Big Data

1st Pétala G. da S. E. Tuy
IME. Federal University of Bahia
Salvador, Brazil
petalatuy@hotmail.com

2nd Tatiane Nogueira Rios
IME. Federal University of Bahia
Salvador, Brazil
tatiane.nogueira@gmail.com

***Abstract*—The performance of Fuzzy Rule-Based Classification Systems (FRBCSs) is highly affected by the increasing number of instances and attributes present in Big Data. Previously proposed approaches try to adapt FRBCSs to Big Data by distributing data processing with the MapReduce paradigm, by which the data is processed in two stages: Map and Reduce. In the Map stage, the data is divided into multiple blocks and distributed among processing nodes that process each block of data independently. In the Reduce stage, the results coming from every node in the Map stage are aggregated and a final result is returned. This methodology tackles vertical high dimensionality (high number of instances), but it does not approach datasets with simultaneous vertical and horizontal high dimensionality (high number of attributes), as it is the case of text datasets. In this work, we deal with the aforementioned drawbacks by proposing *Summarizer*, an approach for building reduced feature spaces for horizontally high dimensional data. To this end, we carry out an empirical study that compares a well-known classifier proposed for vertical high dimensionality datasets with and without the horizontal dimensionality reduction process proposed by *Summarizer*. Our findings show that existing classifiers that tackles vertical Big Data problems can be improved by adding the *Summarizer* approach to the learning process, which suggests that an unified learning algorithm for datasets with a high number of instances as well as a high number of attributes might be possible.**

***Index Terms*—Dimensionality reduction, Fuzzy C-Means, Big Data.**

I. Introduction

Among the wide range of challenges that arise with Big Data are classification tasks, where knowledge is extracted from data to predict future patterns. An expressive part of Big Data is in the format of text that can be used to solve various real life problems, such as spam detection [1], author identification [2], web pages classification [3] and sentiment analysis [4]. Besides the increasing number of instances and attributes present in Big Data, uncertainty, vagueness and noise inherent to such data can mask the valorous information to be obtained from Big Data, making it more difficult to perform classification tasks.

Fuzzy Rule-Based classification Systems (FRBCSs) improve classification and decision support systems by using overlapping class definitions, which enriches the classifier structure, supports the decision-making process, and enables interpretable results [5]. However, FRBCSs are not able to cope with the new challenges that come with Big Data problems.

To use FRBCSs in Big Data scenarios, researchers have been proposing distributed approaches that make use of the MapReduce paradigm and the Apache Hadoop computing framework [6]–[9].

One recent approach, proposed by Elkano et al. (2017a) and named CHI-BD, consists of training the algorithm proposed by Chi et al. (1996) [10] along with multiple nodes, where portions of data are stored and processed in each node in the *Map* stage. Later, the partial results are aggregated and the final result is obtained in the *Reduce* stage. Elkano et al. (2017a) defend that rules quality are not affected by the degree of parallelism and, therefore, CHI-BD outperforms the previous one in terms of runtime and classification performance when dealing with Big Data problems. The existing adaptations of FRBCSs have shown that applications of fuzzy rule-based systems are promising alternatives to deal with vertically high dimensional datasets (high number of instances), as it is the case of CHI-BD. However, the performances of the algorithms with simultaneous vertical and horizontal (increasing number of attributes) high dimensional datasets were not explored.

CHI-BD was tested and validated in datasets with millions of instances, but the larger tested dataset had a maximum of 54 attributes. It is well known that some datasets, such as text datasets, may have thousands of attributes. In such type of data, only distributing the data in multiple mappers may not overcome the high dimensionality problem, since each mapper will have a reduced number of instances but will still have a very high number of attributes. In a Rule-Based System, the number of attributes directly affects the number of rules since each attribute is a component in every rule. That is, the more attributes there is in the dataset, the more components each rule will have and, consequently, the more rules will be necessary to compose the System. Systems with high number of rules will have its interpretability and accuracy negatively affected [11], [12]. For this reason, systems with smaller number of rules are preferable.

Other than MapReduce, researches have used training set selection (TSS) to reduce dimensionality of data and improve classification results. Similar to the MapReduce approaches cited above, as the CHI-BD algorithm, TSS only tackles vertical high dimensionality. That is, TSS approaches reduce the number of instances, but the number of attributes remain high. [13]–[15].

978-1-7281-6932-3/20/$31.00 ©2020 IEEE

Feature selection techniques can be used to tackle this problem by selecting a smallest subset of attributes [16], [17]. However, the existing feature selection processes might not guarantee a fair representation of all classes by the selected features, specially for imbalanced datasets. These techniques evaluate each attribute individually, and dependencies between attributes might be ignored, resulting on the selection of redundant attributes and on the discard of relevant ones [18], [19].

In this work, we deal with the aforementioned drawbacks by clustering the attributes of such datasets to reduce horizontal high dimensionality. Consequently, classification performance may be improved by using such groups of attributes as the new feature space. It is expected that the smaller feature space will represent the data properly and that the classification algorithms will present good classification results in terms of accuracy. It is also expected that a smaller number of rules will be generated since a smaller number of features will be used, which will result in more interpretable systems. Therefore, with our approach, a smaller feature space is built for each dataset using the well-known fuzzy clustering algorithm, Fuzzy C-Means (FCM) [20].

In order to evaluate the performance of our approach, we have compared 9 binary text classification datasets in terms of accuracy and Rule Base size, when using FCM and two other well-known dimensionality reduction techniques: Principal Component Analysis (PCA) [21], and Latent Semantic Analysis (LSA) [22]. The performance of CHI-BD using the original datasets was then compared to CHI-BD using the reduced feature spaces.

The structure of this work is given as follows. In Section II, we summarize some basic concepts of the MapReduce Paradigm and give a brief description of the CHI-BD algorithm. In Section III, we briefly describe the three approaches used for dimensionality reduction in this work: FCM, PCA, and LSA. Section IV presents the new framework to deal with horizontally and vertically high dimensional datasets. The experimental study and the analysis of the results are shown in Section V. Finally, the conclusions of this work are presented in Section VI.

II. Vertical dimensionality reduction

Classification tasks over datasets with vertical high dimensionality have been successfully addressed by means of the MapReduce paradigm. MapReduce is a distributed programming model proposed by Google in 2004 [23] with the goal of simplifying the distributing process. Distributed programming is a strategy that distributes the computation flow along large-scale clusters of machines. The MapReduce model divides the computational flow in two main stages: Map and Reduce. Each stage is organized around key-value pairs. The Map and Reduce functions are described bellow.

Map function: in this stage, the data is automatically divided into independent data blocks and distributed along storage nodes. The data is analyzed independently by each node, and, for each input, intermediate results are returned by each node.

Reduce function: in this stage, the results returned by each node in the Map function are collected and aggregated to produce a final result.

One good approach that makes use of the MapReduce paradigm is CHI-BD [24], which is a more robust version of Chi et al.'s (1996) algorithm, dealing with vertically high dimensional datasets. CHI-BD is divided in three stages, where each stage is composed of a MapReduce process. The flow of the algorithm was summarized here bellow. Further details about the algorithm flow can be found in Elkano et al. (2017).

1) **Rules generation process**
   In this stage, a preliminary rule base is built ignoring rule weights. It is composed of the following MapReduce process:
   a) *Map stage*
      i) Data is split into different mappers
      ii) In each mapper, one rule is created for each example
   b) *Reduce stage*
      i) Rules with the same antecedent are grouped and the list of all possible consequents is returned.
      ii) A new rule base is created with antecedents and the list of consequents.
2) **Frequent subsets search**
   This stage aims to avoid repeated computations when calculating rule weights. This way, the following MapReduce process find subsets of antecedents that appear most often in the rule base built in the rules generation process.
   a) *Map stage*
      i) The antecedent part of each rule is divided into a definite number of subsets of contiguous antecedents.
      ii) A key-value pair is generated, where *key* is a tuple with the subset and its id, and *value* is 1.
   b) *Reduce stage*
      i) The number of occurrences of each subset is counted, and the subsets having a number of occurrences larger than a given threshold are kept.
3) **Computation of rule weights**
   In this stage, rule weights are computed and included in the preliminary rule base created in stage 1.
   a) *Map stage*
      i) Each mapper loads the rule base previously obtained.
      ii) In each mapper, one rule is created for each example
         A) For each rule, the matching degrees of all examples related to each possible class of the rule is computed.

b) *Reduce stage*

i) The matching degrees in all mappers are summed and used to compute the rule weights.

The algorithm presented above is the most recent adaptation of the Chi et al.'s algorithm to the Big Data scenario. This adapted approach shows that fuzzy rule-based systems can be easily implemented in parallel processing environments.

III. Horizontal Dimensionality reduction

The proposed approach aims to reduce horizontal high dimensionality by creating groups of attributes as a new feature space of the dataset to be classified. To this end, the FCM algorithm was chosen to perform the clustering step. In addition to FCM, two established dimensionality reduction techniques were applied: Principal Component Analysis (PCA) and Latent Semantic Analysis (LSA). Both approaches use Singular Value Decomposition (SVD) and leverage the idea that meaning can be extracted from context. In LSA, context is extracted by means of the term×document matrix, while in PCA context is provided through the term covariance matrix. Details about the above mentioned techniques are given below.

A. Fuzzy C-Means

Clustering algorithms are unsupervised techniques that aim to identify *groups* of similar instances [25]. In a crisp approach, each instance belongs to a specific group of similar instances, while in a fuzzy approach each instance belongs to every group with different membership degrees. Fuzzy C-Means (FCM) [20] is the most popular fuzzy clustering algorithm and was considered in this work due to its simplicity and effectiveness.

Distance and similarity are key concepts for constructing clustering algorithms, since groups are built based on similarity and distance measurements. A similarity function $Sim()$ computes similarity degrees between instances, whose values are in the range [0,1]. Consider a set of n examples $D = \{d_1, d_2, \ldots, d_n\}$, where d_k, $k = 1, \ldots, n$, is a m-dimensional point. Usually, if $Sim(d_i, d_j) > Sim(d_i, d_l)$, where $i, j, l \in k$, we can say that instance d_j is more similar to instance d_i if compared to instance d_l. The Cosine similarity measure is commonly used in the context of text processing due to the high dimensionality of the instances to be compared [26]. The Cosine similarity between two instances is defined as follows.

$$Sim(d_i, d_j) = cos(\theta) = \frac{\vec{d_i} \cdot \vec{d_j}}{||\vec{d_i}|| \cdot ||\vec{d_j}||} \tag{1}$$

where $\vec{d_i}$ and $\vec{d_j}$ are instances in the vector form and θ is the angle between the two vectors.

For FCM, an example can belong to all groups with different membership degrees. Membership degrees are associated to the distance of the examples to the centroids of the groups. The more distant is the example to the centroid of some group, the smaller is its membership degree concerning to that group. A fuzzy clustering is composed by a set of c groups, denoted by $P = \{A_1, A_2, \ldots, A_c\}$, and a partition matrix $W = w_{k,p} \in [0, 1]$, for $p = 1, \ldots, c$, where each element $w_{k,p}$ represents the membership degree of the example k in the group A_p [27].

The sum of all membership degrees for a given example d_k must be equal to 1, as shown in Equation 2.

$$\sum_{i=1}^{c} w_{k,p} = 1 \tag{2}$$

Each group A_p must contain at least one example with non-zero membership degree and must not contain all the points with membership degrees equal to 1, as shown in Equation 3.

$$0 < \sum_{k=1}^{n} w_{k,p} < n \tag{3}$$

The goal of the FCM algorithm is to minimize the sum of the squared error (SSE), as shown in Equation 4. That is, the goal is to minimize the distances between the examples and the centroids c_p of the groups.

$$SSE = \sum_{k=1}^{n} \sum_{i=1}^{c} w_{k,p}^{f} dist(d_k, c_p)^2 \tag{4}$$

where $f > 1$ is the fuzzifier parameter that can influence the performance of the FCM algorithm.

B. Principal Components Analysis (PCA)

PCA is a mathematical matrix decomposition technique for dimensionality reduction. It reduces dimensionality by finding linearly uncorrelated attributes that minimize information loss and maximize the explained variance of the data. It was first proposed by Karl Pearson [21] and has been widely used since then [28].

The steps for performing PCA on a $n \times m$ dimensional dataset is given as follows:

- **Data standardization**: All m attributes are transformed to the same scale.
- **Covariance matrix computation**: The $m \times m$ covariance matrix is built to represent the relationship between all the attributes.
- **Eigenvectors and eigenvalues computation**: The m eigenvectors and eigenvalues of the covariance matrix are computed.
- **Feature vector construction**: The $c \ll m$ principal components are selected, and the $n \times c$ final dataset is represented by the principal components.

Further data analysis techniques can be performed with the smaller dataset represented by Principal Components.

C. Latent Semantic Analysis (LSA)

LSA is a technique usually used for natural language processing to analyze relationships between terms and documents [22]. Its main idea is to filter noise and reduce dimensionality by finding the smallest set of concepts that explain all the

documents, where concepts are patterns of words that usually appear together in documents.

LSA takes the term×document matrix and applies the Singular Value Decomposition (SVD) technique to obtain the reduced data space [29]. SVD is a factorization of a $n \times m$ matrix, D, into three components USV^T. Where U is an $n \times c$ orthogonal matrix, whose columns are defined by the left-singular vectors of D. S is a $c \times c$ diagonal matrix, V is a $n \times c$ matrix, with V^T being the transpose of V. The columns of V are defined by the right-singular vectors of D. The value $c \ll m$ is called the rank and defines the number of singular values that are to be kept.

The number of singular values in the diagonal matrix S defines the amount of variance explained by each of the singular vectors, and it is used to define the new reduced dimension of the transformed data matrix.

Once FCM, PCA and LSA can be used to reduce horizontal high dimensionality, next we present our proposal, named *Summarizer*, which makes use of vertical and horizontal dimensionality reduction, as previously explained, to improve classification accuracy.

IV. Summarizer

The general idea of our proposal includes adding a clustering step before the classification process, by which the attributes of a dataset will be groups of attributes obtained through the FCM algorithm. A crisp clustering also could be considered, but a fuzzy clustering was chosen to use as much information as a dataset can have, since using FCM each attribute can be present in every cluster to some extent.

By reducing the high number of attributes to a much smaller number of groups, there will be much less horizontal high dimensionality of the data and then a classification task can be performed, as for example, CHI-BD.

To better formulate the process of creating groups of attributes, consider a dataset with n instances and m attributes. The matrix of data is represented in Table I.

TABLE I
DATA MATRIX ($I \times A$).

Instance	**Attribute 1**	**Attribute 2**	...	**Attribute** m
Instance 1	d_{11}	d_{12}	...	d_{1m}
Instance 2	d_{21}	d_{22}	...	d_{2m}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Instance n	d_{n1}	d_{n2}	...	d_{nm}

where d_{ij} is the value of the i-th instance and the j-th attribute, with $i = 1, \ldots, n$ and $j = 1, \ldots, m$. The first step for creating groups of attributes is to transpose the data matrix, so an attribute $\times$ instance ($A \times I$) matrix is obtained. The transposed matrix is presented in Table II.

The next step of the process is to apply FCM on the attributes with varying quantities of groups. The output in this step will be an attribute $\times$ group ($A \times G$) matrix, represented in Table III.

TABLE II
TRANSPOSED DATA MATRIX ($A \times I$).

Attribute	**Instance 1**	**Instance 2**	...	**Instance** n
Attribute 1	d_{11}	d_{21}	...	d_{1n}
Attribute 2	d_{12}	d_{22}	...	d_{2n}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Attribute m	d_{m1}	d_{m2}	...	d_{mn}

TABLE III
ATTRIBUTE $\times$ GROUP MATRIX ($A \times G$).

Attribute	**Group 1**	**Group 2**	...	**Group** c
Attribute 1	w_{11}	w_{12}	...	w_{1c}
Attribute 2	w_{21}	w_{22}	...	w_{2c}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Attribute m	w_{m1}	w_{m2}	...	w_{mc}

Here, w_{jl} represents the degree of membership of the attribute j to the group l, with $j = 1, \ldots, m$ and $l = 1, \ldots, c$, where $c \ll m$. Since the goal is to associate instances to groups, matrix $A \times G$ will be converted to an instance $\times$ group matrix ($I \times G$). This conversion is straight forward since every instance is associated to all attributes and every attribute is associated to all groups. This way, the $I \times G$ matrix is created by calculating the weighted average between the values of the attributes in every instance and the membership degree of the attributes in the groups. The x_{il} component of the $I \times G$ matrix is calculated as in Equation 5.

$$x_{il} = \frac{\sum_{j=1}^{m} d_{ij} \times w_{jl}}{\sum_{j=1}^{m} w_{jl}} \tag{5}$$

where $i = 1, \ldots, n$, $l = 1, \ldots, c$, and $j = 1, \ldots, m$. Matrix $I \times G$ is represented in Table IV. The Instance $\times$ Group matrix ($I \times G$) is the data in the final format to be used for the classification task with the desired algorithms.

TABLE IV
INSTANCE $\times$ GROUP MATRIX ($I \times G$).

Instance	**Group 1**	**Group 2**	...	**Group** c
Instance 1	x_{11}	x_{12}	...	x_{1c}
Instance 2	x_{21}	x_{22}	...	x_{2c}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Instance n	x_{n1}	x_{n2}	...	x_{nc}

The above mentioned steps is summarized as follows:

1) Transposing the original data matrix (instance $\times$ attribute) in order to get an attribute $\times$ instance matrix.
$$I \times A \rightarrow A \times I$$
2) Cluster the attributes with different number of groups. The output in this step will be an attribute $\times$ group matrix.
$$A \times I \rightarrow A \times G$$

3) Calculating the weighted averages for each instance in each group to get the instance $\times$ group matrix.

$$A \times G \rightarrow I \times G$$

4) Perform classification task using groups as attributes.

Summarizer can be used in any classification system for any type of data, since its main idea is to reduce the feature space of a dataset without loss of information. To check its feasibility when Big Data is considered, in this work, the experiments were performed for text classification tasks using the CHI-BD algorithm. Therefore, the experiments were performed as shown in the workflow in Figure 1: in Step 1, by means of FCM, PCA, and LSA, an horizontal dimensionality reduction is performed; and in Step 2, by means of CHI-BD, a vertical dimensionality reduction is performed for a classification task.

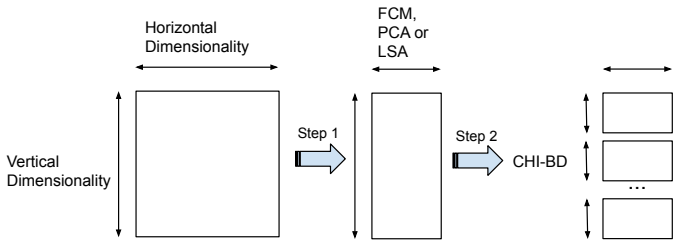

Fig. 1. Workflow of *Summarizer*

The results obtained from our experiments are presented in the next section.

V. Experimental Results

This study is aimed at assessing the performance of our proposal (*Summarizer*) by analyzing the results obtained in terms of classification accuracy and number of rules. We compare the results obtained through FCM, PCA and LSA algorithms to the results of CHI-BD with no clustering step. Section V-A describes the datasets, the parameters, and the statistical tests considered for the study. In sections V-B and V-C we analyze the results in terms of number of rules and classification performance.

A. Experimental Framework

To conduct the experiments, five high dimensional datasets were selected from the LABIC website[1]. In order to obtain more binary text datasets, we have turned multi-class classification problems into multiple binary One-vs-All problems, which resulted in 9 binary text datasets. To do so, we selected a positive class and considered the rest of the classes as the negative class. Descriptions of the datasets are shown in Table V with the number of examples (#Examples), the number of examples of majority and minority classes (#maj;#min), and the number of attributes (#Attributes).

TABLE V
Summary of the datasets.

Dataset	#Examples	(#maj;#min)	#Features
20ng_0	1000	(500;500)	8526
20ng_1	2000	(1500;500)	11027
20ng_2	2000	(1500;500)	11027
20ng_3	2000	(1500;500)	11027
20ng_4	2000	(1500;500)	11027
Ohscal	2881	(1621;1260)	8214
Re8	6215	(3923;2292)	7846
Multidomain sentiment	8000	(4000;4000)	13360

The experiments were conducted applying a 5-fold cross validation. Therefore, the result of each dataset with each technique was computed as the average of the accuracies in each fold. Larger datasets were not considered in this work since an optimized version of FCM would be necessary for clustering larger feature spaces. We are considering parallel grouping approaches to enable dealing with even larger datasets is an intended future work.

The parameters considered for executing CHI-BD were the default parameters available in the original implementation of the algorithm[2]. It was considered three linguistic labels per attribute, and it was applied the winning rule fuzzy reasoning method for classifying examples. Rule weights have been computed using the *Penalized Cost-Sensitive Certainty Factor* (PCF-CS) [7], which is an adaptation of the *Penalized Certainty Factor* (PFC) [30]. To conduct the experiments it was used an Ubuntu virtual machine Version 16.04.1 with 256 GB of RAM memory and 32 cores, operating on a VirtualBox 5.2.1 platform under a machine equipped with an Intel(R) Xeon(R) CPU E7-2890 v2 processor at 2.80GHz.

The performance of the *Summarizer* technique will be assessed by means of geometric means (Gmeans) [31], a well known metric for imbalanced datasets, and also by means of the number of rules.

The FCM algorithm was used in the horizontal dimensionality reduction step to create groups of attributes varying from 2 and 10 groups to proportions of 1%, 5% and 10% of the number of attributes. The groups of attributes were then used as new attributes for the CHI-BD algorithm, which configures the *Summarizer* approach. The different number of groups was then compared to the traditional approach (with no horizontal dimensionality reduction). The horizontal dimensionality reduction step was also performed using PCA and LSA under the same conditions. The results were also compared to the traditional approach and to the FCM approach in terms of accuracy and GM.

The comparisons in terms of number of rules generated by CHI-BD for each dataset and for each method is discussed in the next section.

B. Number of Rules

Figure 2 shows the number of rules generated by CHI-BD without *Summarizer* for each of the 9 datasets considered in

[1] $http://sites.labic.icmc.usp.br/text\_collections/$

[2] https://github.com/melkano/CHI-BD

this work, as well as the number of rules generated by CHI-BD for different variations of *Summarizer* (2 and 10 groups, generated by FCM, PCA and LSA, to proportions of 1%, 5% and 10% of the number of attributes).

As an example of rule generated by the model, consider the rule generated for the dataset Re8, with *Summarizer* with FCM and 2 groups. The rule is shown bellow.

IF Group1 IS L_2 AND Group2 IS L_2 THEN CLASS = $class_1$
WITH RW = 0.7408715

Rules or concepts/groups interpretation was not addressed in this work, and it is a suggestion for future assessment. However, the smaller number of components in each rule is an indicator that rules interpretation can be improved by *Summarizer*.

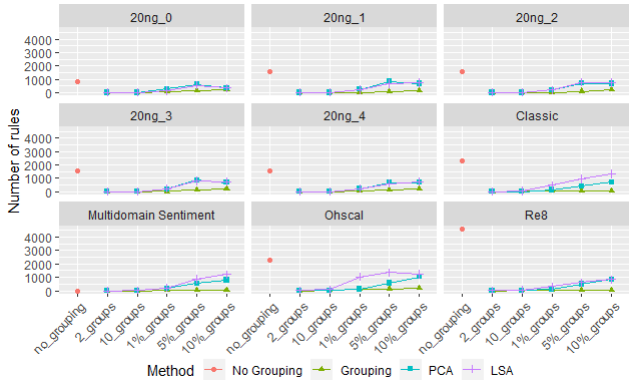

Fig. 2. Number of rules generated by the CHI-BD algorithm.

According to the results in Figure 2, the number of rules generated by CHI-BD with no grouping step (without *Summarizer*) was higher for all datasets when compared to *Summarizer* with FCM, PCA and LSA. The number of rules for CHI-BD with no grouping step for the Multidomain Sentiment dataset could not be counted because the number of rules was to high to be written in the Rule Base file. Among the *Summarizer* approaches used with CHI-BD, FCM returned the smaller number of rules when compared to PCA and LSA for all datasets.

It can be observed that, for same datasets, the number of rules generated by CHI-BD decreases for *Summarizer* with 10% of the number of attributes. That can be an indication that redundancy starts being added to the model when a high number of groups of terms is considered. This way, a smaller number of groups may carry all the information about the data and adding more groups only increases dimensionality with no extra information gain.

The average number of rules for every number of groups generated by *Summarizer* is presented in Table VI. The average number of rules generated by CHI-BD without *Summarizer* was equal to 1813. The percentage of decrease on the number of rules obtained by CHI-BD with *Summarizer* compared to CHI-BD without *Summarizer* (1813) is also presented in table VI (percentages in parenthesis). In order to find statistical differences between the number of rules, the Wilcoxon test [32] was applied. The p-values are also presented in Table VI.

As can be seen in Table VI, among FCM, PCA and LSA, *Summarizer* with FCM presented the smaller average number of rules. For *Summarizer* with 10% of the number of terms, for example, the average number of rules for FCM was equal to 147 rules, while the average number of rules was equal to 722 and 888 for *Summarizer* with PCA and LSA, respectively. That is, with the same ammount of attributes (groups), CHI-BD with *Summarizer* considering FCM was capable of designing a classification system with a much smaller number of rules.

Comparisons of the methods in terms of Gmeans will be presented in next section, and one should keep in mind that there should be a balance between classification accuracy and rule base size, since the number of rules in a classification system play an important role in the interpretability of the results.

TABLE VI
P-VALUES, AVERAGE NUMBER OF RULES FOR CHI-BD WITH *Summarizer* AND ITS PERCENTAGE OF DECREASE WHEN COMPARED TO CHI-BD WITHOUT *Summarizer*

Summarizer *Number of groups*	**p-value**	**Average Number of Rules** *FCM*	*PCA*	*LSA*
Summarizer 2	< 0.0001	3 (99.8%)	4 (99.8%)	5 (99.7%)
Summarizer 10	< 0.0001	10 (99.4%)	15 (99.2%)	28 (98.4%)
Summarizer 1%	< 0.0001	48 (97.4%)	192 (89.4%)	349 (80.8%)
Summarizer 5%	< 0.0001	108 (94.0%)	636 (64.9%)	804 (55.6%)
Summarizer 10%	< 0.0001	147 (92.0%)	722 (60.2%)	888(51.0%)

The number of rules for CHI-BD with FCM, PCA and LSA drops at a high rate for all approaches of *Summarizer* when compared to CHI-BD without *Summarizer*. The percentage of decrease on the number of rules of *Summarizer* with FCM was above 90% for all number of groups of terms. Based on the results, it can be observed a significant reduction on the number of rules of CHI-BD with *Summarizer* when compared to CHI-BD without *Summarizer* with a significance level of 99% (p-value < 0.0001) in all cases.

Despite the significant decrease on the number of rules for CHI-BD with *Summarizer*, there was not a negative impact on the performance of the algorithm, as discussed in the next section.

C. Classification Performance

Table VII shows the Gmeans values obtained by CHI-BD with and without *Summarizer* for each of the 9 datasets considered in this work. The best overall result for each dataset is shown underlined, while the best one for each dataset and for each number of groups on *Summarizer* among FCM, PCA and LSA is shown in bold-face.

As shown in Table VII, for 8 out of the 9 studied datasets, the performance of CHI-BD with *Summarizer* was better than the performance of CHI-BD without *Summarizer* in terms of Gmeans for at least one of the horizontal dimensionality reduction approaches (FCM, PCA or LSA). For most of the cases, CHI-BD with *Summarizer* with only 2 groups

TABLE VII
GMEANS FOR CHI-BD WITH AND WITHOUT *Summarizer*.

		Gmeans		
Dataset	**Method**	*FCM*	*PCA*	*LSA*
$20ng_0$	No horizontal reduction		.7071	
	Summarizer 2	.7051	.7195	**.7232**
	Summarizer 10	**.7121**	.7029	.7089
	Summarizer 1%	.6071	**.7367**	.7361
	Summarizer 5%	.5689	.6570	**.7293**
	Summarizer 10%	.5881	.7060	**.7071**
$20ng_1$	No horizontal reduction		**.7500**	
	Summarizer 2	.4755	.4917	.4654
	Summarizer 10	.5006	.5005	.5005
	Summarizer 1%	.3567	.3198	.5466
	Summarizer 5%	.4779	.1885	.4018
	Summarizer 10%	.4890	.1000	-
$20ng_2$	No horizontal reduction		-	
	Summarizer 2	**.5004**	.4620	.5064
	Summarizer 10	**.5018**	.5015	.5015
	Summarizer 1%	.3007	.5312	**.5336**
	Summarizer 5%	.2718	**.2946**	.2604
	Summarizer 10%	**.2575**	-	-
$20ng_3$	No horizontal reduction		0.1000	
	Summarizer 2	.1684	**.3484**	.1912
	Summarizer 10	.1013	**.2940**	.1000
	Summarizer 1%	.2901	.5071	**.5203**
	Summarizer 5%	.2961	.3028	**.3229**
	Summarizer 10%	**.3612**	-	.0000
$20ng_4$	No horizontal reduction		.1540	
	Summarizer 2	.1142	.2925	**.4106**
	Summarizer 10	**.4778**	-	.1278
	Summarizer 1%	**.4916**	.3324	.3342
	Summarizer 5%	**.5082**	.3590	.3924
	Summarizer 10%	**.4926**	.1414	-
$Classic_0$	No horizontal reduction		.7147	
	Summarizer 2	.6802	**.9751**	.8888
	Summarizer 10	.7073	**.9670**	.9173
	Summarizer 1%	.7080	**.8278**	.8076
	Summarizer 5%	**.7343**	.2528	.7300
	Summarizer 10%	**.7277**	.4726	.7163
Ohscal	No horizontal reduction		.7501	
	Summarizer 2	.5549	.6560	**.8124**
	Summarizer 10	.6707	.3621	**.8147**
	Summarizer 1%	.7712	.3703	**.7934**
	Summarizer 5%	.6795	.6787	.7317
	Summarizer 10%	.7246	.7497	.7488
Re8	No horizontal reduction		.8034	
	Summarizer 2	.3696	.7945	.7979
	Summarizer 10	.2774	**.8296**	.8013
	Summarizer 1%	.2726	.7910	**.8077**
	Summarizer 5%	.4337	.7989	.8022
	Summarizer 10%	.7199	.7942	.7949
Multi-domain sentiment	No grouping		.7071	
	Summarizer 2	.6825	**.7072**	.3850
	Summarizer 10	.3136	**.7072**	.6968
	Summarizer 1%	.0641	.7060	.7035
	Summarizer 5%	.2620	.2437	.3502
	Summarizer 10%	.5792	.7071	**.7075**

of attributes already results in better Gmeans than CHI-BD without *Summarizer*.

For most of the datasets, it can be found a percentage of reduction on the number of terms that results in an increase on the Gmeans metric for some of the FCM, PCA and LSA method in comparison to CHI-BD without *Summarizer*. It should be noted that there was a significant reduction on the number of rules for CHI-BD with *Summarizer* in addition to the increase of Gmeans. The best possible method would be the one with the higher Gmeans and the smaller number of rules.

Table VIII presents a comparison between the number of times CHI-BD with *Summarizer* had a better Gmeans performance than CHI-BD without *Summarizer*. The results are presented in terms of wins (W), which is the number of times the use of some reduction approach obtained a classification result superior to the CHI-BD without *Summarizer*; ties (T), which is the number of times the use of some reduction approach obtained a classification result similar to the CHI-BD without *Summarizer*; and loses (L), which is the number of times the use of some reduction approach obtained a classification result inferior to the CHI-BD without *Summarizer*. Finally, the Total row indicates the total number of wins, ties and loses considering all possible number of groups for *Summarizer*.

A total column was added to Table VIII in order to consider the number of wins, ties and loses when any of the three approaches presents a better Gmeans than CHI-BD without *Summarizer*. For example, a win is considered in the Total column when FCM or PCA or LSA presented a better Gmeans than CHI-BD without *Summarizer*.

TABLE VIII
GMEANS FOR CHI-BD WITH AND WITHOUT *Summarizer*

Summarizer	**Gmeans W/T/L**			
Number of groups	Total	*FCM*	*PCA*	*LSA*
Summarizer 2	7/0/2	3/0/6	7/0/2	7/0/2
Summarizer 10	8/0/1	4/0/5	5/0/4	4/1/4
Summarizer 1%	7/0/2	4/0/5	5/0/4	7/0/2
Summarizer 5%	5/0/4	4/0/5	3/0/6	5/0/4
Summarizer 10%	5/1/3	4/0/5	1/1/7	3/1/5
TOTAL	32/1/12	19/0/26	23/1/21	26/2/17

As shown in Table VIII, despite the much smaller number of rules obtained by CHI-BD with *Summarizer*, it still gives a higher Gmeans value in 32 out of 45 cases (71.1% of wins). LSA presented the higher number of wins (26 wins). For the Total column, *Summarizer* with 10 groups presented the higher number of wins (8 out of 9 datasets), followed by *Summarizer* 2 and 1% with 7 wins.

VI. CONCLUSIONS

As exposed by Elkano et al. (2019) [33], in the last few years, researchers have tried to design fuzzy classifiers that reduce memory and computational requirements on Big Data classification tasks. However, these algorithms often consider Big Data as datasets with large number of instances, with no concern about the high number of attributes as well. In this work, we have proposed a new approach for building reduced feature spaces of vertical and horizontal high dimensional datasets. We tested our approach with CHI-BD, that already deals with vertical high dimensionality by making use of the MapReduce paradigm. Our method was proven to significantly reduce the number of rules generated by CHI-BD, since a much smaller feature space is considered instead of the original set of attributes. Despite the significant reduction on the number of features, and consequently on the number of rules generated by the model, there was not a negative impact on the performance of the algorithm in terms of Gmeans.

This work led us to many possibilities of future work. First, we consider developing a less empirical approach for defining the parameters of the model. Second, we also consider testing a MapReduce approach on the process of building the reduced

feature space. Third, rules and concepts/groups interpretation is also being considered as a future work, since much smaller rules are generated by *Summarizer*. Finaly, we consider testing *Summarizer* with algorithms other than CHI-BD for dealing with vertical high dimensionality.

REFERENCES

[1] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, "6 million spam tweets: A large ground truth for timely twitter spam detection," in *IEEE International Conference on Communications (ICC)*. London, UK: IEEE, 2015, pp. 7065–7070.

[2] J. M. B. Silva and F. Silva, "Feature extraction for the author name disambiguation problem in a bibliographic database," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: ACM, 2017, pp. 783–789.

[3] E. Saraç and S. A. Özel, "An ant colony optimization based feature selection for web page classification," *The Scientific World Journal*, vol. 2014, 2014.

[4] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093 – 1113, 2014.

[5] H. Ishibuchi, T. Nakashima, and M. Nii, *Classification and modeling with linguistic information granules: Advanced approaches to linguistic Data Mining*. Springer Science & Business Media, 2006.

[6] T. White, *Hadoop: The definitive guide*. " O'Reilly Media, Inc.", 2012.

[7] V. López, S. del Río, J. M. Benítez, and F. Herrera, "Cost-sensitive linguistic fuzzy rule based classification systems under the mapreduce framework for imbalanced big data," *Fuzzy Sets and Systems*, vol. 258, pp. 5–38, 2015.

[8] M. Elkano, M. Galar, J. Sanz, and H. Bustince, "Chi-bd: A fuzzy rule-based classification system for big data classification problems," *Fuzzy Sets and Systems*, 2017.

[9] S. del Rio, V. Lopez, J. M. Benitez, and F. Herrera, "A mapreduce approach to address big data classification problems based on the fusion of linguistic fuzzy rules," *International Journal of Computational Intelligence Systems*, vol. 8, no. 3, pp. 422–437, 2015.

[10] Z. Chi, H. Yan, and T. Pham, *Fuzzy algorithms: with applications to image processing and pattern recognition*. World Scientific, 1996, vol. 10.

[11] H. Ishibuchi and T. Yamamoto, *Trade-off between the Number of Fuzzy Rules and Their Classification Performance*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 72–99.

[12] Y. Jin, "Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 2, pp. 212–221, 2000.

[13] N. Verbiest, J. Derrac, C. Cornelis, S. García, and F. Herrera, "Evolutionary wrapper approaches for training set selection as preprocessing mechanism for support vector machines: Experimental evaluation and support vector analysis," *Applied Soft Computing*, vol. 38, pp. 10–22, 2016.

[14] G. Acampora, F. Herrera, G. Tortora, and A. Vitiello, "A multi-objective evolutionary approach to training set selection for support vector machine," *Knowledge-Based Systems*, vol. 147, pp. 94–108, 2018.

[15] S. Garcia, J. Derrac, J. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 3, pp. 417–435, 2012.

[16] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, p. 94, 2017.

[17] M. Labani, P. Moradi, F. Ahmadizar, and M. Jalili, "A novel multivariate filter method for feature selection in text classification problems," *Engineering Applications of Artificial Intelligence*, vol. 70, pp. 25–37, 2018.

[18] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[19] P. Kumbhar and M. Mali, "A survey on feature selection techniques and classification algorithms for efficient text classification," *International Journal of Science and Research*, vol. 5, no. 5, p. 9, 2016.

[20] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. USA: Kluwer Academic Publishers, 1981.

[21] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[22] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse processes*, vol. 25, no. 2-3, pp. 259–284, 1998.

[23] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[24] M. Elkano, M. Galar, J. Sanz, G. P. Dimuro, and H. Bustince, "A global distributed approach to the chi et al. fuzzy rule-based classification system for big data classification problems," in *Fuzzy Systems (FUZZ-IEEE), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1–6.

[25] A. Fahad, N. Alshatri, Z. Tari, A. Alamri, I. Khalil, A. Y. Zomaya, S. Foufou, and A. Bouras, "A survey of clustering algorithms for big data: Taxonomy and empirical analysis," *IEEE transactions on emerging topics in computing*, vol. 2, no. 3, pp. 267–279, 2014.

[26] A. Strehl, J. Ghosh, and R. Mooney, "Impact of similarity measures on web-page clustering," in *Workshop on artificial intelligence for web search (AAAI 2000)*, vol. 58, 2000, p. 64.

[27] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic*. Prentice hall New Jersey, 1995, vol. 4.

[28] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.

[29] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," in *Linear Algebra*. Springer, 1971, pp. 134–151.

[30] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems," *IEEE transactions on fuzzy systems*, vol. 13, no. 4, pp. 428–435, 2005.

[31] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 463–484, 2011.

[32] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

[33] M. Elkano, H. Bustince, and M. Galar, "Do we still need fuzzy classifiers for small data in the era of big data?" in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, June 2019, pp. 1–6.