IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 30, 2019, accepted July 17, 2019, date of publication July 22, 2019, date of current version August 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2930355

KSUFS: A Novel Unsupervised Feature Selection Method Based on Statistical Tests for Standard and Big Data Problems

JOSÉ A. SÁEZ AND EMILIO CORCHADO
Department of Computer Science and Automatics, University of Salamanca, 37008 Salamanca, Spain
Corresponding author: José A. Sáez (joseasaezm@usal.es)

ABSTRACT The typical inaccuracy of data gathering and preparation procedures makes erroneous and unnecessary information to be a common issue in real-world applications. In this context, feature selection methods are used in order to reduce the harmful impact of such information in data analysis by removing irrelevant features from datasets. This research presents a novel feature selection method in the field of unsupervised learning, in which the complexity arises from the fact that the class labels cannot be used to select the most discriminative features as it is traditionally performed in supervised learning. The technique designed, which is called *Kolmogorov-Smirnov test-based Unsupervised Feature Selection* (KSUFS), is based on the computation of estimated feature distributions that are later compared to the original ones using non-parametric statistical tests to provide the most representative input variables. Two versions of the KSUFS are presented in this study: one of them is particularly designed to deal with standard data, in which the accuracy of the method prevalences over other of its aspects; the other version is designed to treat with big data problems, in which the computational complexity is improved due to the characteristics of this type of datasets. The KSUFS is successfully compared to other state-of-the-art unsupervised feature selection techniques in a thorough experimental study, which considers both standard and big data problems. The results obtained show that the method proposed is able to outperform the rest of reference unsupervised feature selection methods considered in the comparisons, selecting the first most influential features for standard datasets and particularly highlighting when big data problems are treated.

INDEX TERMS Big data, clustering, feature selection, statistical tests, unsupervised learning.

I. INTRODUCTION

The complexity of data preparation processes in real-world applications, such as those related to medicine [26] or big data processing [4], usually produces datasets containing unnecessary and erroneous information [30], [39], [45]. Features incorporating such harmful information may imply important drawbacks in data analysis [27]. These problems include, among others, the increase of the computational and memory requirements of learning algorithms, as well as the deterioration of their performance and, therefore, the interpretability of the models built from the data [31], [35].

In order to overcome the aforementioned problems, *dimensional reduction* methods [7], [34] are used to preprocess datasets aiming to find a subset of relevant features to reduce their dimensionality. Two types of dimensional reduction techniques are distinguished [6], [36]: feature extraction and feature selection. *Feature extraction* methods [7], [43] transform the original variable space to perform dimensional reduction. *Feature selection* [6], [34] chooses a subset of features from a given dataset removing its irrelevant and noisy features in order to represent the original data. Between these two approaches, this research focuses on feature selection methods because many applications require of building highly-interpretable models [26], [35] and, therefore, the meaning of the original variables in the data must be retained.

In supervised problems [27], [44], in which each one of the observations in the dataset has an associated class label, feature selection methods usually evaluate the importance of variables according to the discriminant information encoded in the classes [8], [15]. On the other hand, *unsupervised*

The associate editor coordinating the review of this manuscript and approving it for publication was Bilal Alatas.

feature selection [10], [38], [46] is generally a more difficult problem due to the lack of class labels in the dimensionality reduction process. Since supervised feature selection is not directly applicable to these data, the design of new feature selection approaches for unsupervised datasets is of a great interest. Among them, *filter methods* [24], [28] highlight by using different criteria to perform feature selection in a time-efficient way and without using any learning algorithm to estimate the quality of the features.

The study of unsupervised feature selection methods proposed in the literature shows that, even though there are methods that do not need to be aware of the number of clusters in a dataset to provide a final solution [17], [46] (which are called *non-cluster-aware* methods hereinafter), other techniques do require of a user parameter specifying the number of clusters in the problem [28], [37] (which are called *cluster-aware* methods). Note that cluster-aware methods do not need the exact number of clusters in which the data are grouped to work, but their performance is optimized when it is provided [37]. This information, approximate though, may imply an advantage in favor of cluster-aware approaches. However it is usually unknown, especially in early stages of data analysis when preprocessing methods, such as feature selection, are applied.

Due to the aforementioned facts, this research proposes a novel technique for unsupervised feature selection that does not use the information of cluster cardinality in the data and it is based on the potential of statistical tests to best identify the most influential features. First, the proposal is based on estimating a value distribution for each feature in the dataset using the rest of the data available employing the well-known *k-Nearest Neighbors Imputation* (KNNI) [3], which analyzes the relationships among features to determine such estimations. Then, non-parametric statistical tests [32] are employed to set the significance of each variable for unsupervised learning by comparing its original and estimated distributions, which posibilites the final feature selection. The main aim of this comparison is to detect and remove those features having the highest changes between their estimated and original distributions, which are those variables more likely to contain unnecessary and erroneous information since they are not predictable using the rest of the data. The *Kolmogorov-Smirnov* statistical test [33], which is used to evaluate such comparison between distributions, gives name to the proposal: *Kolmogorov-Smirnov test-based Unsupervised Feature Selection* (KSUFS).

The idea of using non-parametric statistical tests to evaluate feature importance in datasets was previously researched in other works [29]. However, unlike this study that focuses on unsupervised problems, the aforementioned research was oriented to deal with classification problems, in which class label information was available for the computations and methods employed. Furthermore, the approach of [29] was designed to work as a feature weighting scheme to be included as a part of the distance function in the *Nearest Neighbor* (NN) [23] classifier. Therefore, instead of finding a subset of features as the approach of this research does, all the features were simultaneously considered and weighted inside the distance function of NN in [29]. Thus, KSUFS selects a subset of features in unsupervised problems using internal operations that do not consider the class information and being independent of the learning algorithm used later.

Difference from existing works in the specialized literature on unsupervised feature selection, this research deeply analyzes in a novelty way two different scenarios in which datasets with a very differentiated dimensionality in terms of the number of features are considered. Thus, Section VI focuses on the application of feature selection in *standard* unsupervised problems, which are commonly used in the data mining literature and in which the number of features can be handled by learning algorithms in a moderate time. On the other hand, Section VII studies feature selection with *big data* [16] unsupervised problems, having a *high-dimensionality*, which represents an interesting scenario to apply unsupervised feature selection since it may help to reduce the higher time and storage computational requirements of learning algorithms used later. Accordingly, this research proposes two alternative versions of KSUFS, depending on the characteristics of the problem addressed: one of these versions is designed to work with standard data (Section III), whereas the other version is specially designed to work with high-dimensional big data problems (Section IV). The study of unsupervised feature selection divided into two different scenarios (standard and big data), let us to check whether the necessities of problems with different sizes are the same from the point of view of feature selection and whether the behavior of the feature selection methods studied changes when considering each type of data.

A thorough empirical study will be developed comparing 8 representative unsupervised feature selection techniques [24], both non-cluster-aware and cluster-aware, to KSUFS. All of them will be used to preprocess 20 standard real-world datasets and 20 big data real-world problems (having more than 250 features) taken from the UCI repository [9]. Such preprocessing will consider 6 different percentages of features to be selected, from 15% to 90% by increments of 15%. Finally, following other related works [17], [46], the well-known *k-means* [41] clustering algorithm will be used to estimate the quality of the feature selection performed in each preprocessed dataset. Such outcomes, which will imply more than 44000 performance results, will serve as a solid basis to establish a comparison among the different feature selection methods considered, not applying any preprocessing and the proposal of this work. The conclusions reached will be based on the usage of the appropriate statistical tests [12], [40] in order to check the significance of the differences found. The main lessons learned in this research, including interesting findings and conclusions related to the experimentation performed and its analysis are summarized in Section VIII. Full results and details of the experimentations

are available in the web-page associated to this research at https://joseasaezm.github.io/ksufs/.

The rest of this study is organized as follows. Section II introduces feature selection in unsupervised problems. Section III presents the details of KSUFS when working with standard datasets, whereas Section IV focuses on the peculiarities of the method dealing with big data. Section V describes the experimental framework. Then, Section VI analyzes the results obtained from the comparison with other models considering standard data, whereas Section VII focuses on the results with big data problems. Section VIII summarizes the lessons learned. Finally, Section IX enumerates some concluding remarks and future research lines.

II. UNSUPERVISED FEATURE SELECTION

Feature selection [6], [34], [47] is a preprocessing technique whose main aim is to determine a subset of variables from a given dataset removing its irrelevant and noisy features. Such dimensionality reduction eases the subsequent discovering of meaningful patterns hidden in the data [20]. In unsupervised feature selection, where there are not class labels available, different criteria to remove those variables that are not interesting for data analysis are used [19], [46]. As it was previously mentioned, unsupervised feature selection can be broadly divided into cluster-aware and non-cluster-aware methods [17], [37], depending on they use or not the number of clusters in the data as a parameter.

Within non-cluster-aware methods, some algorithms solve the feature selection problem calculating a score for each feature and selecting those with the best value of such measure. *Laplacian Score Feature Selection* (LSFS) [14] and *Maximum Variance-based Feature Selection* (MVFS) [24] are two well-known methods following this scheme. LSFS evaluates the importance of each feature according to the *Laplacian score*. This method aims at preserving the similarity to the original data space selecting those features that retain sample locality, which is determined by an affinity matrix. MVFS ranks the features in the dataset based on their variances and selects the top K features corresponding to the maximum values in order to obtain the best expressive power.

Closely related to LSFS, *Spectral Feature Selection* (SPEC) [17] considers the interactions among features in order to select a subset of variables preserving the manifold structure of the original data. SPEC establishes comparisons to the eigenvectors of a normalized Laplacian matrix to select features that assign similar values to instances that are related according to the affinity matrix. Other methods, such as *Similarity Preserving Feature Selection* (SPFS) [46], also address the intrinsic relationships among variables for a feature selection preserving the similarity to the original data sample but, moreover, include a sparse multiple-output regression formulation to improve the effectiveness in the optimization problem to find the final solution.

On the other hand, cluster-aware methods try to benefit from knowing the number of clusters and exploring the discriminative information in the data to perform a more accurate feature selection. For example, *Multi-Cluster Feature Selection* (MCFS) [37] selects features using a spectral regression with l_1-norm regularization trying to capture the multi-cluster structure of the data, focusing on the correlations between different features. There are cluster-aware methods that simultaneously exploit the discriminative information and consider the interactions between features [19], [42]. For example, *Unsupervised Discriminative Feature Selection* (UDFS) [42] selects features computing an score based on local discriminative information and the manifold structure of the data. *Nonnegative Discriminative Feature Selection* (NDFS) [19] selects features by a joint framework which considers the usage of an $l_{2,1}$ regularized regression and a nonnegative spectral analysis [19], which reduces the effect of noisy features. Other techniques also focus on performing a feature selection robust to noisy variables, such as *Robust Unsupervised Feature Selection* (RUFS) [28], which simultaneously uses an $l_{2,1}$ regularized regression and an $l_{2,1}$ norm-based nonnegative matrix factorization with local learning in the feature selection process. Thus, some of these works show the relevance of reducing the effect of noisy features to perform a more accurate feature selection [19], [28].

TABLE 1. Feature selection methods used in experiments.

Non-cluster-aware unsupervised feature selection		
Method	Ref.	Parameters
LSFS	[14]	Affinity matrix: binary weights; $k = 5$ neighbors
MVFS	[24]	Optimization function: variance maximization
SPEC	[17]	Optimization function: $\hat{\varphi}_1$; Affinity matrix: binary weights; $k = 5$
SPFS	[46]	Optimization function: SFS
Cluster-aware unsupervised feature selection		
Method	Ref.	Parameters
MCFS	[37]	Affinity matrix: binary weights; $k = 5$ neighbors
NDFS	[19]	$\gamma = 10E8$; $\alpha = 1$; $\beta = 1$; Affinity matrix: binary weights; $k = 5$
RUFS	[28]	Affinity matrix: binary weights; $k = 5$ neighbors
UDFS	[42]	$\gamma = 0.1$; Affinity matrix: binary weights; $k = 5$ neighbors

Table 1 summarizes the unsupervised feature selection methods considered in the experiments of this research along with the configuration of their main parameters. These techniques have been chosen because they are well-known representatives and are usually considered as reference methods in the field of unsupervised feature selection applying different strategies to select the most relevant variables. Note that, even though almost all of the parameters are the default ones considered by the authors of such methods, an affinity matrix with binary weights and $k = 5$ neighbors is used for those methods that require it (LSFS, SPEC, MCFS, NDFS, RUFS and UDFS) to establish a fair comparison among them.

From the two aforementioned groups of feature selection, non-cluster-aware and cluster-aware methods, the method presented in this research, KSUFS, belongs to the former group. It gets closer to real-world scenarios in which the number of clusters in the data is unknown. Furthermore, those important aspects treated in the literature on

unsupervised feature selection, such as considering the interactions between variables to determine the quality of features [17], [46] and reduce the importance of noisy features [19], [28], are also included in the proposal. All the details about the working of KSUFS and how it addresses the above issues are detailed in Sections III-IV, presenting the peculiarities of the approach to treat with standard datasets and the modifications necessary to adapt KSUFS for big data, respectively.

III. A NOVEL UNSUPERVISED FEATURE SELECTION BASED ON THE KOLMOGOROV-SMIRNOV STATISTIC

This section describes the main foundations of the proposed feature selection method for unsupervised problems, called *Kolmogorov-Smirnov test-based Unsupervised Feature Selection* (KSUFS). KSUFS, which has been programmed under Java, can be used with data of any dimensionality, both standard datasets and big data problems.

The working of the method with standard data, which do not have a very large number of features and can be addressed by learning algorithms in a reasonable time, is explained in this section. Considering standard data, exact computations (with independence of their computational cost) are used in KSUFS to ensure the maximum accuracy of the method.

Big data [5], [16] is defined as high volume, velocity and/or variety data requiring of new processing techniques to be addressed. Thus, big data problems are not identified by specific size measures, but by the fact that traditional methods are not able to work with them due to their volume, velocity or variety. In order to take advantage of the knowledge contained in big data, the developing of techniques capable of handling such data is of a great interest. Thus, when dealing with high-dimensional big data, the computational complexity, i.e. the velocity, of the methods is as important than their performance [18]. Therefore, it is particularly interesting to adapt KSUFS to improve its computational cost dealing with big data with respect to the version to deal with standard data.

KSUFS in either of its versions, for standard and big data, is based in three main steps (see Figure 1). Additionally, these steps are reflected in the pseudocode shown in Algorithm 1. Let $\varphi_1, \ldots, \varphi_m$ ($|\varphi_i| = n, \forall i \in \{1, \ldots, m\}$) be the features belonging to an unsupervised dataset with n examples and m variables, the steps of KSUFS are:

1) *Step 1 - Creation of estimated feature distributions*. The first step of KSUFS computes, for each feature φ_i in the input dataset D, an estimated distribution of values φ'_i ($|\varphi'_i| = n$) conforming an estimated data D'.
2) *Step 2 - Comparison of real and estimated feature distributions*. This step uses a non-parametrical statistical test, the *Kolmogorov-Smirnov* test [33], to compare the distribution of the values for each feature φ_i of D and the corresponding estimated feature φ'_i of D'. This enables the computation of the D^i_n statistic for each feature φ_i, which ranks each feature according to its importance in the unsupervised problem.
3) *Step 3 - Feature selection according to statistic values*. The original features φ_i are ordered from the lowest value (the most representative variables) to the highest value of their D^i_n statistic computed in the previous step. This procedure leads to a new set of ordered features denoted as φ''_j ($|\varphi''_j| = n/j = 1, \ldots, m$). Finally, the $K (K \leq m)$ most representative features are selected to be part of the final output dataset D''.

The steps of KSUFS to deal with standard data are detailed from Section III-A to III-C, whereas Section IV describes the modifications to work with big data problems.

Algorithm 1 Pseudocode of the KSUFS Method

Input: original dataset D, selected features K.
Output: processed dataset D''.

Step 1 - Creation of estimated feature distributions

Set the estimated dataset $D' = \emptyset$;
for *each feature* $\varphi_i \in D$ **do**
  for *each example* $\delta_j \in D$ **do**
    Suppose the feature value $\varphi_{i,j}$ as missing;
    $\varphi'_{i,j} \longleftarrow$ Estimate $\varphi_{i,j}$ using the rest of D;
  end
  $D' \longleftarrow D' \cup \{\varphi'_i\}$;
end

Step 2 - Comparison of real and estimated distributions

for *each feature* $\varphi_i \in D$ *and* $\varphi'_i \in D'$ **do**
  $D^i_n \longleftarrow$ Compute the *Kolmogorov-Smirnov* statistic using φ_i and φ'_i;
end

Step 3 - Feature selection according to statistic values

$L \longleftarrow$ Sort features φ_i in increasing order of D^i_n;
Set $\varphi''_1, \ldots, \varphi''_m$, with $\varphi''_j = \varphi_i / D^i_n$ in j-th place of L;
$D'' \longleftarrow$ Select the first K features φ''_j;

A. STEP 1 - CREATION OF ESTIMATED FEATURE DISTRIBUTIONS FROM THE ORIGINAL DATA

In order to build the distribution φ'_i for each feature φ_i in the data, KNNI [3] is used. This is an imputation method [11], [13], [25] used to analyze the relationships among features and set a missing value in the data. Imputation methods have shown a good performance estimating missing values from the rest of the data, so they are also suitable for use in KSUFS. Furthermore, they are independent of the learning algorithms used later, so one may choose the most adequate imputation method that, in our case, is adapted to the characteristics of unsupervised problems.

KNNI is based on computing the k-nearest neighbors of a sample of a dataset characterized with a missing value and, then, averaging the values of the feature with the miss-

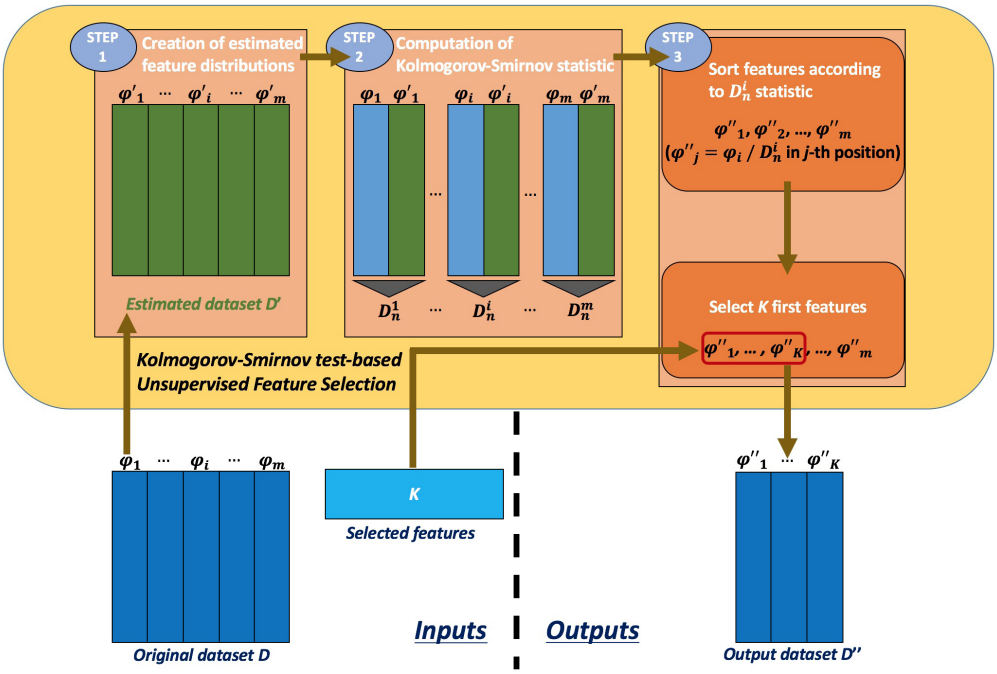


FIGURE 1. Unsupervised feature selection proposed.

ing value for these neighbors to compute the final solution. Thus, KNNI does not require of a complex initialization of parameters to work properly. In the experiments of this research, KNNI uses the *Euclidean distance* as a similarity function and $k = 10$, such as it is normally used in the literature [3]. This technique offers several advantages for unsupervised feature selection compared to other imputation methods:

1) *Need of class labels to make estimations.* Due to limitations of working with unsupervised problems, the imputation method cannot use information from class labels. Difference from other well-known methods, such as *Support Vector Machine Imputation* (SVMI) [11] or *Concept Most Common* (CMC) [13], KNNI does not need class labels to estimate missing values.
2) *Consideration of feature interactions to make estimations.* The imputation of KNNI for a feature value considers its interactions with the rest of features. Thus, the distribution of imputed values for each feature φ'_i is conditioned to all other features. Some methods, such as *Global Most Common* (GMC) [13] or CMC, do not take advantage from the interactions among variables to estimate feature values. Note that this issue, that is, considering the relationships among variables, it is important in unsupervised feature selection [42].

The Step 1 of KSUFS (lines 1-8 in Algorithm 1) creates a new estimated dataset D' from the original one D using KNNI. If the original data D is composed of the features $\varphi_1, \varphi_2, \ldots, \varphi_m$, the imputed data D' will be formed by the features $\varphi'_1, \varphi'_2, \ldots, \varphi'_m$ whose values are obtained by KNNI.

The procedure to obtain D' from D is represented in Figure 2. This is based on assuming iteratively that each feature value $\varphi_{i,j}$ of each example δ_j of the dataset D is missing and estimating a new value $\varphi'_{i,j}$ for it using KNNI. Thus, after selecting the target feature φ_i and its corresponding target feature value $\varphi_{i,j}$ to be estimated (tasks 1-2 in Figure 2), this value is supposed to be missing. KNNI is then used to predict a new value for that feature value (task 3). KNNI computes the k-nearest neighbors $\delta_{\eta_1}, \ldots, \delta_{\eta_k}$ of the example δ_j using all the features except φ_i, i.e. using $\varphi_j / j \neq i$. The new value $\varphi'_{i,j}$ associated to $\varphi_{i,j}$ is computed as the average of the feature values in φ_i of these neighbors (task 4):

$$\varphi'_{i,j} = \frac{\sum_{q=1}^{k} \varphi_{i,\eta_q}}{k}$$

The new estimated dataset D' is obtained by repeating this process for each feature value $\varphi_{i,j}$, until the whole original dataset D has been processed. Carrying out this process, it is possible to estimate a distribution of values for each feature, which is conditioned to the rest of the features. In this way, KSUFS considers the interactions between variables to later determine the quality of each feature, as it is recommended in the specialized literature [17], [46]. The new dataset D' will contain these conditioned distributions for each feature. This will allow us to check those features that are the most difficult to predict using the rest of the variables and, therefore, they are likely to contain unnecessary and erroneous information making them less important to the unsupervised task.

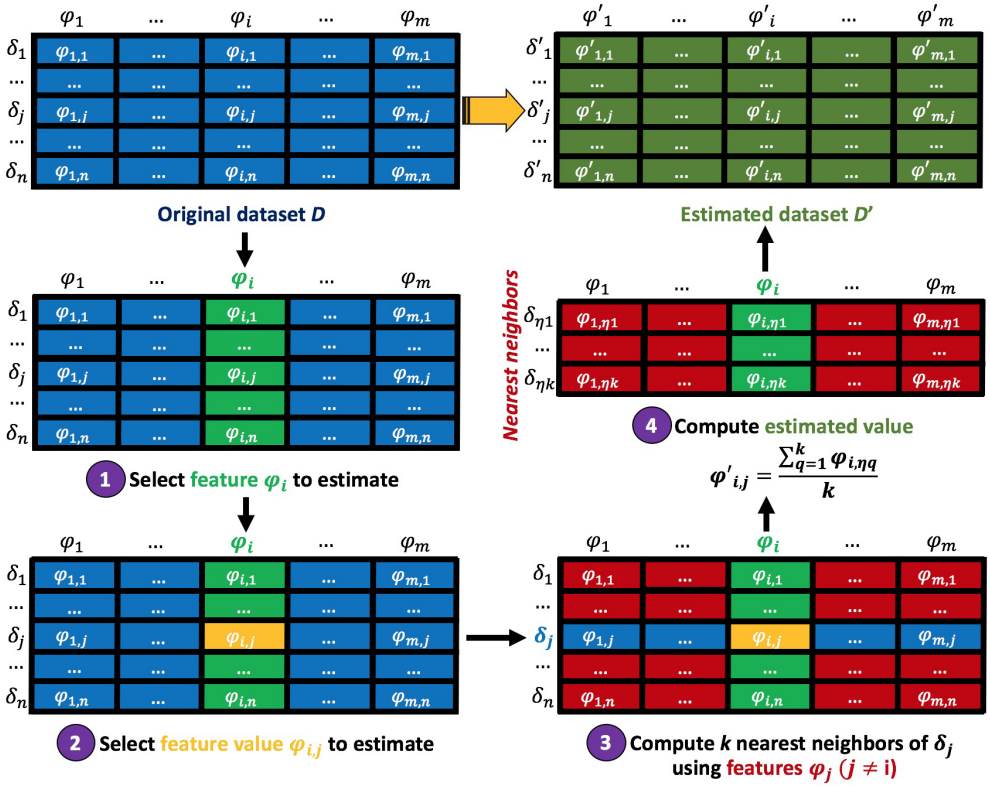


FIGURE 2. Creation of estimated feature distributions from the original ones using KNNI.

B. STEP 2 - COMPARISON OF REAL AND ESTIMATED FEATURE DISTRIBUTIONS USING STATISTICAL TESTS

After the Step 1, a new estimated conditioned distribution φ'_i is obtained for each original feature φ_i. The Step 2 (lines 9-11 in Algorithm 1) consists of comparing, for each feature, their distributions φ_i and φ'_i, checking in this way the degree of change in each one of the features after the application of the imputation method. Some features may present small changes between their distributions of values φ_i and φ'_i, being usually more easily predictable using the rest of the data, whereas other features may have a greater degree of change between their two distributions.

In order to quantify the degree of the changes between φ_i and φ'_i, the *Kolmogorov-Smirnov* test [33] is used. This test computes a statistic D_n, which can be regarded as a measure of how different two samples are. One of its main advantages is that it let us to quantify the distance between the empirical distribution functions of two samples, concretely those implying φ_i and φ'_i and, furthermore, without making any assumption about the probability distributions of the variables since it is a non-parametric test. The null distribution of its statistic, D_n, is computed under the null hypothesis that the samples are drawn from the same distribution.

For our purpose consisting of measuring the similarity of two given distributions, shape metrics, such as D_n, are more appropriate than other simpler statistics used to compare distributions. Let us consider, for example, the case of the variance as an instance of dispersion metrics. Variance measures how the feature values are grouped around the mean. However, this represents only one of the characteristics to analyze when comparing distributions. D_n contains structural information describing the changes in the distribution and identifying the parts of the feature domain with higher or lower concentrations of values. This fact shows the potential of the D_n statistic against other simpler metrics, such as the variance, to capture changes in feature distributions.

Thus, let X and Y two samples and F_X and F_Y their empirical distribution functions:

$$F_X(x) = \frac{1}{n}\sum_{h=1}^{n} I_{X_h \leq x}, \quad F_Y(x) = \frac{1}{n}\sum_{h=1}^{n} I_{Y_h \leq x}$$

where $I_{X_h \leq x}$ is the indicator function, equal to 1 if $X_h \leq x$ and equal to 0 otherwise. The *Kolmogorov-Smirnov* statistic is shown in Equation 1:

$$D_n = \sup_x |F_X - F_Y| \in [0, 1] \tag{1}$$

Table 2 shows an example of the computation of the *Kolmogorov-Smirnov* statistic D_n from two distributions $X = \{X_1, \ldots, X_n\}$ and $Y = \{Y_1, \ldots, Y_n\}$ with $n = 5$.

The D_n^i statistic is, therefore, computed for each pair of distributions $X = \varphi_i$ and $Y = \varphi'_i$, $\forall i \in \{1, \ldots, m\}$. By quantifying the distance between the empirical distribution functions of the two samples involved in each comparison (φ_i and φ'_i),

TABLE 2. Example of computation of the *kolmogorov-smirnov* statistic D_n in two samples X and Y of size $n = 5$. In this case, $D_n = \sup_x |F_X - F_Y| = 0.6$.

$X = \{0.12, 0.29, 0.44, 0.68, 0.94\}$
$Y = \{0.02, 0.11, 0.18, 0.26, 0.63\}$

x	F_X	F_Y	$\lvert F_X - F_Y \rvert$
0.00	0	0	0
0.02	0	0.2	0.2
0.11	0	0.4	0.4
0.12	0.2	0.4	0.2
0.18	0.2	0.6	0.4
0.26	0.2	0.8	0.6
0.29	0.4	0.8	0.4
0.44	0.6	0.8	0.2
0.63	0.6	1	0.4
0.68	0.8	1	0.2
0.94	1	1	0

each of these statistics can be interpreted as a measure of how different the two samples are.

We must remark that the *Kolmogorov-Smirnov* test is used in KSUFS to compute its D_n^i statistic, which let us compare the real and estimated value distributions for each one of the features. This statistic will serve to rank the features according to their importance for the unsupervised task and make the final feature selection in Step 3. Note that the computation of the D_n^i statistic is one of the stages used by the *Kolmogorov-Smirnov* test and it does not imply applying the full statistical test, which would finally produce a p-value.

C. STEP 3 - UNSUPERVISED FEATURE SELECTION ACCORDING TO STATISTIC VALUES

The values of D_n^i can be used to rank each feature φ_i according to its importance for feature selection. Inspired by works showing the benefits of reducing the relevance of unnecessary and noisy variables [19], [28], KSUFS reduces the effect that changing features according to the estimations of KNNI have, that is, those usually having a higher D_n^i.

Before explaining the reasoning behind reducing the effect of features with a high D_n^i, let us clarify the relationship between the quality of the estimations of KNNI (in Step 1) and the obtained values of D_n^i (in Step 2). For each original feature $\varphi_i = \{\varphi_{i,1}, \cdots, \varphi_{i,n}\}$, KNNI estimates a value for each one of the n examples in the data, resulting in $\varphi_i' = \{\varphi_{i,1}', \cdots, \varphi_{i,n}'\}$. Thus, both φ_i and φ_i' have an implicit association of their values, that is, in their j-th value $\varphi_{i,j}'$ is the estimation of $\varphi_{i,j}$ for the example δ_j. On the other hand, the *Kolmogorov-Smirnov* test is used to compare φ_i and φ_i' as distributions. In this case, the aforementioned association between the $\varphi_{i,j}'$ and $\varphi_{i,j}$ is not considered to compute D_n^i, and the distributions φ_i and φ_i' are treated by themselves, without assuming any order in their values. This results in two main situations when analyzing the relationship between the estimations of KNNI and the values D_n^i:

1) KNNI *performs good approximations to the real values.* If the estimated values $\varphi_{i,j}'$ by KNNI are close to the real ones $\varphi_{i,j}$ for each example δ_j, the real and estimated distributions, φ_i and φ_i', will be necessarily quite similar. This situation will produce low D_n^i values.
2) KNNI *performs bad approximations to the real values.* If the estimation $\varphi_{i,j}'$ for each example δ_j is quite different from the real value $\varphi_{i,j}$, two cases are distinguished:
   a) Since $\varphi_{i,j}'$ is quite different from $\varphi_{i,j}$ for each example δ_j, the first and most likely case is that the real and estimated distributions, φ_i and φ_i', to be dissimilar enough too, which will produce high values of D_n^i.
   b) In the second case the estimated distribution φ_i', although it is composed by wrong predictions of KNNI, is by chance similar to the real distribution φ_i. This will produce low values of D_n^i. Note that this case could occur because of the association between the estimation $\varphi_{i,j}'$ and the real value $\varphi_{i,j}$ for each example δ_j is not considered when computing D_n^i.

Thus, a feature φ_a with a low D_n^a value due to this second case b) could be confused with another feature φ_b with a low D_n^b value corresponding to good estimations of KNNI. However, note that this fact, although it is possible, is hard to occur in practice and to affect significantly to the final feature selection. First, for the confusion between φ_a and φ_b in the feature selection process to have a notable impact on results, it requires of much worse estimations for φ_a by KNNI than those for φ_b. Then, by chance, the estimated distribution φ_a' must fit the real distribution φ_a better than the φ_b' distribution fits φ_b (being φ_b and φ_b', by definition, similar enough).

Therefore, in general terms from a practical point of view, and considering the above exception of the second case b), a relationship between good estimations of KNNI and low values of D_n^i and worse estimations of KNNI and higher values of D_n^i can be established.

In this way, due to the presence of larger amounts of errors in noisy features and the absence of correlation with other variables of unnecessary features, they are probably wrongly predicted by KNNI. Because of this, as it was previously stated, the distributions of values φ_i and φ_i' are more likely to be different, implying a higher D_n^i value. Under this consideration, we aim to highlight the effect that features with a small D_n^i, since they are more likely to conform a block of key features describing the dataset.

In Step 3 (lines 12-14 in Algorithm 1), all the features φ_i are finally ordered from lowest to highest of their corresponding D_n^i statistic. This results in a set of ordered features $\varphi_1'', \varphi_2'', \ldots, \varphi_m''$, being $\varphi_j'' = \varphi_i / D_n^i$ is in the j-th position of the ordered list of D_n^i statistics. Then, similar to other unsupervised feature selection methods [17], [28], [46], the top first K ($K \leq m$) variables φ_j'' are selected to be part of the final output dataset D''.

IV. ADAPTING THE UNSUPERVISED FEATURE SELECTION IN KSUFS FOR BIG DATA PURPOSES

This section proposes a series of adaptations to KSUFS when dealing with big data problems characterized by having large qualities of features. The main aim of such adaptations is maintaining the basics of the proposal described in Section III reducing the computation costs implied without noticeably affecting the performance of the method.

Section IV-A details the modifications of KSUFS to be adapted to big data problems, whereas Section IV-B analyzes the improvements in its computational complexity with respect to the version to treat with standard data.

A. MODIFICATIONS TO DEAL WITH BIG DATA

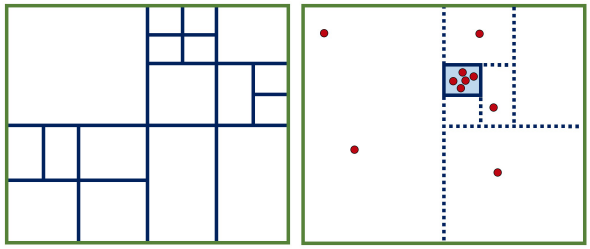
The version of KSUFS to treat with standard data described in Section III is based on three main steps: 1) creation of estimated feature distributions (Section III-A), 2) comparison of real and estimated feature distributions (Section III-B) and 3) feature selection (Section III-C).

As it is shown in Section III, Steps 2 and 3 require no complicated computations and, therefore, the most complex step from the computational point of view is Step 1. Since the last two steps do not strongly affect the complexity of KSUFS, they are maintained without changes in the version of KSUFS for big data. Thus, the Steps 2 and 3 of the version of KSUFS proposed in this section are those presented in Section III-B and III-C, respectively.

The adaptation of KSUFS to big data is therefore focused on improving the computational cost of Step 1 (the creation of estimated feature distributions), which is the most time-consuming when working with this type of problems. Two adaptations are made to this end:

1) *Approximate computation of nearest neighbors*. Dealing with big data, the differences between the set of exact k-nearest neighbors and other sets of approximate nearest neighbors are likely to be minimal. Note that nearest neighbors are used by KSUFS to compute averaged values of their variables to create the estimated feature distributions, which are then compared to the original feature distributions. The impact of computing the exact nearest neighbors may be compensated because all the estimated feature distributions are affected in the same way when considering approximate nearest neighbors and only the relative differences of estimated and original feature distributions are determinant to establish the final ranking of features in KSUFS.

   Thus, the *Approximate Nearest Neighbors* (ANN) [2] method is used instead of KNN in KSUFS for big data. The main idea behind ANN is to preprocess the dataset in such a way that nearest neighbors are computed efficiently. Data are organized using *Balanced Box-Decomposition* (BBD) trees [2], which divide the dimension space and distribute the examples efficiently to save time when computing the nearest neighbors.



(a) Splitting. (b) Shrinking.

FIGURE 3. Examples of (a) multiple iterations of splitting and (b) shrinking, with the dashed lines being the splitting operations and the inner box highlighted.

   In a BBD tree, each node is associated with a region in the domain space. Each region is either an outer rectangular box or the set theoretic difference between an outer rectangular box and an inner rectangular box [2]. The basic building of a BBD tree is done by a sequence of two alternating operations (see Figure 3):

   - *Splitting*: a region is bisected by a hyperplane orthogonal to one of its longest sides. This operation highlights by its simplicity and speed of computation.
   - *Shrinking*: this partitions a region R_u with a box lying inside it, creating an inner subregion R_i. Although the shrinking process has some peculiarities depending on whether the region R_u already has an inner box [1], [2], the essential procedure finds the new inner box R_i by recursively applying splitting operations on the subregion with the larger number of samples. The process stops when the subregion contains no more than $2/3$ of the samples of the initial region R_u. The outer box of this subregion is the inner partitioning box R_i of the shrinking operation.

   As the domain space is being divided, each region is associated with a node in the BBD tree. The root of tree represents all the domain and, thus, all the data samples. Each stage of the building algorithm determines how to subdivide the current region, either through splitting or shrinking, and then partitions the samples among the corresponding child nodes of the tree [1], [2]. Finally, nearest neighbors of a given sample are found traversing the tree and looking for the closest nodes to that containing the query sample as indicated in [2].

2) *Missing values are not assumed when calculating the nearest neighbors*. To create the estimated distribution φ'_i, each feature value $\varphi_{i,j}$ of each example δ_j in the original feature φ_i is iteratively assumed to be missing and a new value $\varphi'_{i,j}$ is estimated using KNNI (see Step 1 in Algorithm 1). To estimate $\varphi_{i,j}$, KNNI computes the k-nearest neighbors of δ_j using all the features except φ_i, since its value $\varphi_{i,j}$ is considered unknown. Under such consideration, the distribution of each feature φ_i is conditioned to the rest of the features, since all the features except φ_i are used when computing the nearest

TABLE 3. Comparison of computational complexity of KSUFS in its three steps and corresponding subparts dealing with standard and big data –being n the number of examples, m the number of features and k the number of nearest neighbors computed to estimate feature distributions.

KSUFS	Standard data	Big data
Step 1		
Structure	✗	$O(m \cdot n \cdot \log n)$
k-NN search	$k \cdot O(m \cdot n)$	$k \cdot O(m \cdot \log n)$
k-NN calls	$m \cdot n$	n
Estimation	$m \cdot n \cdot O(k)$	$m \cdot n \cdot O(k)$
O(Step 1)	$O(m^2 \cdot n^2)$	$O(m \cdot n \cdot \log n)$
Step 2		
KS statistic	$O(n \cdot \log n)$	$O(n \cdot \log n)$
KS calls	m	m
O(Step 2)	$O(m \cdot n \cdot \log n)$	$O(m \cdot n \cdot \log n)$
Step 3		
Ordering	$O(m \cdot \log m)$	$O(m \cdot \log m)$
Selection	$O(1)$	$O(1)$
O(Step 3)	$O(m \cdot \log m)$	$O(m \cdot \log m)$

neighbors of δ_j to estimate $\varphi_{i,j}$. This has the advantage that the feature φ_i does not interfere computing the nearest neighbors determining its estimated values conforming φ'_i. However, the k-nearest neighbors of a given sample δ_j may be different depending on the feature value $\varphi_{i,j}$ assumed to be missing. Thus, this process requires $m \cdot n$ computations of the nearest neighbors in the version of KSUFS for standard data (being n the number of examples and m the number of features).

The second modification of KSUFS in Step 1 for big data is that none of the feature values $\varphi_{i,j}$ of φ_i is considered being missing when computing the nearest neighbors of each example δ_j. In this case, the own feature φ_i interferes calculating the nearest neighbors determining its estimated values of φ'_i, since all the features are considered to compute these nearest neighbors. This modification could affect the fact that the distribution of each feature φ_i is conditioned only to the rest of the features as in the version of KSUFS for standard data. However, the computational cost of KSUFS is improved since, in this case, the nearest neighbors for an example δ_j are only needed to be computed once.

B. ANALYSIS OF COMPUTATIONAL COMPLEXITY

The details of the computational complexity of KSUFS for its three steps in the version to deal with standard data (Section III) and after the adaptations for big data explained in Section IV-A are shown in Table 3.

As it can be appreciated from Table 3, Steps 2 and 3 are those involving a lower computational complexity in the version of KSUFS to deal with standard data. In a dataset containing n examples, Step 2 computes the *Kolmogorov-Smirnov* statistic D_n^i for each attribute $\varphi_i / i = 1, \ldots, m$. Since the computation of D_n^i requires a complexity of $O(n \cdot \log n)$, the overall complexity of this step is $O(m \cdot n \cdot \log n)$. Step 3 focuses on ordering the D_n^i values (implying a complexity $O(m \cdot \log m)$ using the *Quicksort* algorithm) and selecting the top first features, which is $O(1)$. This results in an total complexity for Step 3 of $O(m \cdot \log m)$.

On the other hand, Step 1 of KSUFS presented in Section III is that implying the highest computational cost. This step aims at creating the estimated distribution of values for each feature in the dataset. In order to perform this task and compute each estimated value in such distributions, the *k-Nearest Neighbor* (KNN) algorithm is employed within KNNI. KNN has a complexity of $k \cdot O(m \cdot n)$. Since each attribute value of each example in the dataset is considered to be independently estimated by computing its k-nearest neighbors, KNN is applied $m \cdot n$ times. Finally, the computation of each of the $m \cdot n$ estimated values requires calculating an average value from the k-nearest neighbors, that is, $O(k)$, implying an additional cost of $m \cdot n \cdot O(k)$. Thus, the overall computational complexity of Step 1 is:

$$m \cdot n \cdot k \cdot O(m \cdot n) + m \cdot n \cdot O(k) = O(m^2 \cdot n^2) + O(m \cdot n) = O(m^2 \cdot n^2) \quad (2)$$

The overall computational complexity of KSUFS considering its three steps is, therefore, determined by Step 1, i.e. its complexity is $O(m^2 \cdot n^2)$. The two adaptations of KSUFS to deal with big data detailed in Section IV-A imply a series of improvements in the computational complexity of Step 1:

1) *Approximate computation of nearest neighbors*. The usage of ANN over k-NN requires of creating the *Balanced Box-Decomposition* structure. This structure is built in $O(m \cdot n \cdot \log n)$. Then, in order to compute the k-nearest neighbors of a given example, the computation cost required is $k \cdot O(m \cdot \log n)$. The reader may consult the work of Arya et al. [2] for a detailed analysis of the computational complexity of ANN.
2) *Missing values are not assumed when calculating the nearest neighbors*. Since nearest neighbors are computed once for each example in the dataset, the complexity associated to compute the k-nearest neighbors using ANN for all the examples is $n \cdot k \cdot O(m \cdot \log n)$.

Finally, the computation of each estimated value requires to add a cost of $m \cdot n \cdot O(k)$, which comes from computing the average value of the k-nearest neighbors of the $m \cdot n$ feature values in the dataset. Therefore, the computational cost of Step 1 considering the changes described in Section IV-A is:

$$O(m \cdot n \cdot \log n) + n \cdot k \cdot O(m \cdot \log n) + m \cdot n \cdot O(k) = 2 \cdot O(m \cdot n \cdot \log n) + O(m \cdot n) = O(m \cdot n \cdot \log n) \quad (3)$$

The computational complexity of Step 1 is, thus, improved from $O(m^2 \cdot n^2)$ in the version of KSUFS for standard datasets, to $O(m \cdot n \cdot \log n)$ in the version of KSUFS for big data. This computational-optimized version removes one of the term m in the expression $O(m^2 \cdot n^2)$ and changes another term n by $\log n$, which obviously implies a lower computational cost and faster running times when working with large amounts of features.

TABLE 4. Description of the real-world datasets.

Standard datasets							
Dataset	#exa	#fea	#cla	Dataset	#exa	#fea	#cla
auscredit	690	6	2	miceprotein	552	77	8
bands	365	19	2	pima	768	8	2
biodeg	1055	41	2	saheart	462	8	2
breasttissue	106	9	6	seeds	210	7	3
bupa	345	6	2	seismic	2584	11	2
climatemuq	540	18	2	sonar	208	60	2
german	1000	7	2	wdbc	569	30	2
glass	214	9	6	wine	178	13	3
heart	270	6	2	wisconsin	683	9	2
letter	20000	16	26	wqwhite	4898	11	7
Big data problems							
Dataset	#exa	#fea	#cla	Dataset	#exa	#fea	#cla
allaml	72	7129	2	orl	400	1024	40
arcene	200	10000	2	orlraws10p	100	10304	10
basehock	1993	4862	2	pcmac	1943	3289	2
carcinom	174	9182	11	pixraw10p	100	10000	10
cllsub	111	11340	3	prostate	102	5966	2
coil20	1440	1024	20	relathe	1427	4322	2
glioma	50	4434	4	smkcan	187	19993	2
lung	203	3312	5	tox171	171	5748	4
lymphoma	96	4026	9	usps	9298	256	10
madelon	2600	500	2	yale	165	1024	15

V. EXPERIMENTAL FRAMEWORK

This section presents the details of the experimental framework of this research. Section V-A describes the real-world datasets used in the experimentation. Then, Section V-B explains the methodology for the analysis of the results.

A. REAL-WORLD DATASETS

The experimentation carried out considers 20 standard real-world datasets and 20 high-dimensional big data problems taken from the UCI repository[1] [9]. The standard datasets chosen cover a wide range of cardinalities regarding to the number of examples (from 106 up to 20000), features (from 6 up to 77) and classes (from 2 up to 26). The big data problems selected are characterized by having more than 250 features. They are also different with respect to the number of examples (from 50 up to 9298), features (from 256 up to 19993) and classes (from 2 up to 40).

Table 4 shows a description of all these 40 datasets, along with the number of examples (*#exa*), features (*#fea*) and classes (*#cla*). Note that, even though the datasets employed are supervised problems, class labels are only used to validate the performance of the unsupervised feature selection methods –although the number of classes will be, in fact, used by the cluster-aware methods studied in this research.

B. METHODOLOGY OF ANALYSIS

The unsupervised feature selection methods shown in Table 1 and KSUFS are compared in an experimental study. Because of the nature of all the feature selection methods studied, which rank the variables in the dataset according to a score value to make the final feature subset selection, it is necessary to fix a parameter for the methods indicating the number of features that are desired to be selected in the experiments. In our analysis, 6 different number of features, which will be referred as *control points*, are chosen for each dataset to analyze the effectiveness of each one of the feature selection methods: these range from 15% to 90% of the total amount of features in each dataset, by increments of 15%.

Even though the class labels in the datasets are not used by the feature selection methods, these are employed to estimate their performance using standard *external criteria of quality* [22]. Thus, the *accuracy* (ACC) metric, which is computed as the percentage of observations that are correctly assigned by a clustering method to their corresponding true class labels, is used to evaluate the performance of each method.

As it is derived from the previous definition of accuracy, clustering results are need to compute it by comparing the real class label with the experimental cluster label of each example in a dataset. In order to be able to use this performance criterion, it is necessary to apply a clustering method once the feature selection is performed over the data. Following other related works found in the literature [17], [46], the well-known *k-means* [41] clustering algorithm is used to cluster each dataset with those features given by feature selection techniques. This is an state-of-the-art reference clustering algorithm which has proven to provide good results in a wide variety of problem domains [41]. Finally, before the computation of the performance metrics, the *Kuhn-Munkres* [21] algorithm is used to assign each cluster label to the equivalent label from the dataset.

For each dataset, 20 runs of *k-means* are executed and their performance results are averaged to get stable results. The analysis of results is divided into two different parts, each one of them considering data of different dimensionality:

1) *Study of unsupervised feature selection with standard datasets* (Section VI). The 20 standard datasets in Table 4 are used with all the feature selection methods of Table 1 and our proposal designed to deal with standard datasets (see Section III), which considers the exact computation of the k-nearest neighbors.
2) *Study of unsupervised feature selection with big data* (Section VII). The 20 big data problems will be used with all the feature selection methods and the version of KSUFS for big data (see Section IV).

In the two aforementioned scenarios, the performance of KSUFS is studied considering three different comparisons:

1) *Comparison versus not applying feature selection*. This comparison aims at analyzing if the use of feature selection methods poses an advantage regarding to not preprocessing (denoted as None in the following).
2) *Comparison with non-cluster-aware methods*. This comparison is established between KSUFS and other unsupervised feature selection methods that neither use the number of clusters in the data. These include the methods LSFS, MVFS, SPEC and SPFS.
3) *Comparison with cluster-aware methods*. KSUFS is also compared to other methods that need the number of clusters as a user parameter (MCFS, NDFS, RUFS

[1] http://archive.ics.uci.edu/ml/

and UDFS). The main objective of this comparison is to check if KSUFS offers competitive results against these methods without knowing the amount of clusters.

Any number of clusters may be given to cluster-aware methods, but their performance is usually higher when the real amount is provided [37]. However, although in many situations this number of clusters in unknown, the real number of clusters is used in the experiments of this study with these methods to ensure that they provide competitive results.

The differences between KSUFS and the non-cluster-aware and cluster-aware methods is separately studied motivated by the different nature of both groups. Due to the large amount of results derived from the experimentation –i.e. more than 44000 performance results considering 9 feature selection methods, 40 datasets, 6 control points, the results of None and 20 runs of *k-means* per data– for the sake of brevity only averaged results for the 20 standard and 20 big data problems for each feature selection method and control point are reported. Full results can be found in the web-page of this research. Additionally, the number of datasets in which each method obtains the best result is shown. Please, note that our conclusions are based on the proper statistical analysis, which considers all the results. The following statistical tests are used to analyze the results obtained:

- *Wilcoxon's test* [40]. *Wilcoxon*'s test is applied to study the differences between KSUFS and not using preprocessing (None). The p-values (p_{Wil}) associated with the comparison of the results of the two methods involved over all the datasets will be obtained. They represent the lowest level of significance of a hypothesis that results in a rejection and allow one to know whether two algorithms are significantly different and the degree of their difference. In this research a difference will be considered as significant if the p-value obtained is lower than 0.1. Additionally, the sums of ranks in favor of KSUFS (R^+) and in favor of None (R^-) are shown.
- *Aligned Friedman test* [32] *and Finner procedure* [12]. Regarding the comparison between KSUFS and the other feature selection methods, non-cluster-aware and cluster-aware, the *Aligned Friedman* test [32] will be used. This test will be used to compute the set of ranks that represent the effectiveness associated with each algorithm and the p-value (p_{AF}) of significance of the differences found. In addition, the adjusted p-value p_{Fin} with the *Finner* procedure [12] will be computed.

Note that the above statistical tests are not applied to the samples/features of each individual dataset shown in Table 4, but to the final performance results obtained with *k-means* for the 20 standard datasets or the 20 big data problems. Thus, in each comparison carried out, the corresponding statistical test considers the 20 performance results of the different techniques involved in that comparison.

The aforementioned results, including performances and statistical tests, will be shown in tables of numeric results, which let us represent such information organized properly, being interpretable enough for the analysis of results.

TABLE 5. Accuracy results of KSUFS versus not applying feature selection (None) with standard datasets.

Metric	15%	30%	45%	60%	75%	90%	Best	None
ACC	**60.38**	**62.31**	**61.60**	**60.61**	**60.54**	**59.66**	**63.81**	59.53
$best_{20}$	**10**	**14**	**10**	**10**	9	**13**	**18**	✗
R^+	**121.0**	**158.0**	**112.0**	**95.5**	90.5	**107.5**	**178.0**	✗
R^-	89.0	52.0	78.0	94.5	**99.5**	102.5	12.0	✗
p_{Wil}	5.38E-01	**4.84E-02**	4.81E-01	9.68E-01	8.41E-01	9.11E-01	**2.67E-04**	✗

VI. ANALYSIS OF RESULTS WITH STANDARD DATA

This section focuses on analyzing the performance of the methods with standard datasets. First, Section VI-A considers the application of KSUFS against not preprocessing. Then, Section VI-B shows the comparison of KSUFS with other non-cluster-aware methods, whereas Section VI-C analyzes the results for cluster-aware methods.

A. COMPARISON OF USING FEATURE SELECTION AND NOT PREPROCESSING WITH STANDARD DATASETS

Table 5 shows, for each control point established (15-90%), the accuracy of KSUFS versus not applying any feature selection (None). Note that the results of None are equivalent to select 100% of the features. The column denoted with *Best* considers, for each dataset, the best result obtained among the 6 control points. Table 5 also shows the number of datasets on which KSUFS obtains better results than None (row $best_{20}$) and the results of *Wilcoxon*'s test –the sums of ranks for KSUFS (R^+), for None (R^-) and the p-values (p_{Wil}) associated to the comparison of KSUFS versus None at each control point. The best results at each control point and those significant differences detected with *Wilcoxon*'s tests are highlighted in boldface.

Table 5 shows that the accuracy results of KSUFS always outperforms at each control point to not applying any preprocessing to the datasets. Furthermore, the number of datasets in which KSUFS obtains the best result ($best_{20}$) is always higher for our method at most of the control points –it is only worse than None at the control point 75% and it is tied at the control points 15%, 45% and 60%. The results of *Wilcoxon*'s tests show that KSUFS always obtains more ranks than None (except at the control point 75%) and the associated p-values show that these differences are significant only at the control point 30%. However, when considering the results for the control point with the best performance for each dataset (column *Best*), these clearly show the benefits of feature selection in unsupervised problems, since KSUFS obtains a higher performance than None in almost all the datasets and the differences found are significant, as shown the low p-value obtained (2.67E-04).

These results show the benefits of applying unsupervised feature selection methods with respect to not preprocessing.

TABLE 6. Accuracy results of KSUFS against non-cluster-aware feature selection methods with standard datasets.

Metric	Method	15%	30%	45%	60%	75%	90%
ACC	KSUFS	**60.38**	**62.31**	**61.60**	**60.61**	**60.54**	59.66
	LSFS	57.96	59.75	59.79	60.36	60.29	**59.69**
	MVFS	57.58	59.17	59.28	59.40	59.57	59.57
	SPEC	55.28	56.60	56.05	55.52	57.52	58.66
	SPFS	55.19	58.81	55.97	56.20	59.15	59.54
$best_{20}$	KSUFS	**9**	**8**	7	8	**10**	**9**
	LSFS	6	5	7	**9**	8	7
	MVFS	5	4	**8**	7	6	7
	SPEC	5	6	5	3	3	6
	SPFS	1	3	3	5	6	7
Ranks	KSUFS	**33.53**	**32.65**	**34.13**	**36.23**	**42.23**	46.70
	LSFS	50.48	48.10	44.30	37.98	43.80	**45.03**
	MVFS	52.80	53.55	47.73	46.05	45.93	48.93
	SPEC	58.23	62.68	60.98	69.85	67.35	58.33
	SPFS	57.48	55.53	65.38	62.40	53.20	53.53
p_{AF}	-	**1.84E-03**	**1.88E-03**	**1.94E-03**	**2.06E-03**	**2.18E-03**	**2.06E-03**
p_{Fin}	KSUFS	✗	✗	✗	✗	✗	8.55E-01
	LSFS	**6.47E-02**	**9.22E-02**	2.67E-01	8.49E-01	8.64E-01	✗
	MVFS	**4.72E-02**	**3.02E-02**	1.80E-01	3.60E-01	7.87E-01	7.73E-01
	SPEC	**2.81E-02**	**4.25E-03**	**6.84E-03**	**9.88E-04**	**2.45E-02**	4.71E-01
	SPFS	**2.81E-02**	**2.51E-02**	**2.63E-03**	**8.64E-03**	4.10E-01	5.83E-01

As it is derived from the above analysis of results, KSUFS particularly out-stands over None at the control point 30%, in which *Wilcoxon*'s *p*-values show significant differences with respect to not preprocessing. The results of the approach *Best* should be also remarked, which are very superior to those of None. This approach obtains the best results because it combines the best result for each dataset considering those obtained among all the control points. These results are in favor of *Best* for almost all the datasets (18 out of 20). This fact implies that, for almost all the datasets studied, there exists a percentage of features to be selected, which may vary from one dataset to another, providing a superior performance than not preprocessing and, therefore, feature selection is an useful tool in data analysis processes.

B. COMPARISON WITH NON-CLUSTER-AWARE METHODS USING STANDARD DATASETS

This section analyzes the differences among KSUFS and the rest of the non-cluster-aware methods considered (LSFS, MVFS, SPEC and SPFS). Table 6 shows the accuracy of these techniques. The number of datasets in which each method obtains the best result is also shown (rows $best_{20}$). The results of the *Aligned Friedman* [32] statistical test are depicted in the rows *Ranks* (which represents the sum of ranks in favor of each method) and the row p_{AF}, which represents the *p*-value associated. The best algorithm according to the *Aligned Friedman* test, that is, that with the lowest sum of ranks, is chosen as the control algorithm and compared with the rest of the methods using the *Finner* [12] post hoc procedure, providing the *p*-value associated to each comparison (p_{Fin}).

Analyzing the results in Table 6, one can observe that the accuracy of KSUFS is better than that of the other non-cluster-aware methods, except at the maximum control point 90% in which LSFS gets the best accuracy. The second best method is LSFS, followed by MVFS. Finally, the SPEC and SPFS techniques, with varying results depending on the control point, are found. Regarding to the number of datasets with best result, KSUFS is the best in 4 out of the 6 control points, those with the lowest and the highest percentages of features, although it is competitive at the intermediate control points with the best results of the other methods.

The statistical analysis using the *Aligned Friedman* test provides similar conclusions to the aforementioned ones, being KSUFS the best method for each control point (except at the control point 90%). The very low *p*-values p_{AF} indicate that post-hoc statistical tests can be safely applied to check the significance of the differences found. The post hoc procedure of *Finner* shows that the differences found among the algorithms are significant in favor of KSUFS up to the control point 30%. From this point onwards, the differences detected are only significant with SPEC (up to the control point 75%) and with SPFS (up to the control point 60%) –in the rest of control points and comparisons significant statistical differences are not found.

The above results show the good performance of KSUFS with respect to the rest of non-aware-cluster methods, which are in their same conditions regarding to the knowledge of the number of clusters in the datasets. The good working of KSUFS is especially remarkable in the first control points (up to the control point 30-45%). The differences between KSUFS and some of the other models are smaller from these percentages onwards. This behavior of the feature selection methods could be the expected one dealing with standard datasets. When higher percentages of features are selected, the performance of the different methods is progressively getting close, until all of them reach the same results when 100% of the features are considered. Due to the lower amount of variables in standard data, once a considerable percentage of them have been selected by feature selection methods, i.e. at the intermediate-high control points, it is very likely that most of these techniques share the same features as selected, being their performance more similar. Thus, the most important decisions of feature selection methods on which features to choose dealing with standard data are when determining the quality and order of the first variables. For the aforementioned reasons, those feature selection methods able to out-stand when working with low-medium percentages of features selected in standard data, such as it is the case of KSUFS, are preferable since they show a better behavior selecting the first most influential features.

C. COMPARISON WITH CLUSTER-AWARE METHODS USING STANDARD DATASETS

Table 7 shows the results of comparing KSUFS against the cluster-aware methods, which receive as one of their parameters the exact number of classes in each problem.

TABLE 7. Accuracy results of KSUFS against cluster-aware feature selection methods with standard datasets.

Metric	Method	15%	30%	45%	60%	75%	90%
ACC	KSUFS	**60.38**	**62.31**	**61.60**	**60.61**	60.54	59.66
	MCFS	57.17	59.93	59.69	58.94	60.45	59.67
	NDFS	57.12	59.73	59.97	59.82	59.93	59.83
	RUFS	56.25	58.19	58.17	59.27	**60.66**	58.74
	UDFS	56.36	56.75	57.73	58.36	60.13	**61.12**
$best_{20}$	KSUFS	**9**	**6**	6	**5**	4	**7**
	MCFS	5	**6**	5	**5**	**8**	5
	NDFS	3	**6**	**7**	3	5	**7**
	RUFS	3	4	3	4	6	5
	UDFS	3	3	4	**5**	2	6
Ranks	KSUFS	**32.70**	**32.08**	**36.58**	**44.35**	51.38	48.63
	MCFS	56.88	48.30	46.53	50.75	46.88	56.55
	NDFS	51.58	48.83	47.43	47.98	53.20	45.40
	RUFS	55.98	58.08	59.03	52.23	**45.88**	62.35
	UDFS	55.38	65.23	62.95	57.20	55.18	**39.58**
p_{AF}	-	**1.86E-03**	**1.87E-03**	**1.99E-03**	**2.08E-03**	**2.21E-03**	**2.18E-03**
p_{Fin}	KSUFS	✗	✗	✗	✗	7.74E-01	4.07E-01
	MCFS	**3.32E-02**	**8.95E-02**	3.03E-01	6.29E-01	9.13E-01	1.24E-01
	NDFS	**3.96E-02**	**8.95E-02**	3.03E-01	6.93E-01	7.74E-01	5.25E-01
	RUFS	**3.32E-02**	**9.17E-03**	**2.86E-02**	6.29E-01	✗	**5.12E-02**
	UDFS	**3.32E-02**	**1.21E-03**	**1.61E-02**	5.05E-01	7.74E-01	✗

The accuracies rank as the three best methods to KSUFS, RUFS and UDFS. KSUFS is best technique up to 60% of features, RUFS at the control point 75% and UDFS at 90%. MCFS and NDFS provide similar performance results. Regarding the number of datasets, KSUFS clearly provides the best result at the control point 15% (being the best in 9 out of the 20 datasets). For the rest of the control points, these results are similarly distributed across all the methods –the result of MCFS at the control point 75% is also remarkable.

The *Aligned Friedman* tests shows that KSUFS obtains the best position up to the control point 60% of features selected and it is far away from the other methods up to that level. The control algorithm at the control point 75%, with a lower sum of ranks, is RUFS, whereas UDFS is the best at the control point 90%. The low p_{AF} indicate that the *Finner* test can be applied to estimate the significance of the differences found. Thus, the p_{Fin} values show that there are significant differences in favor of KSUFS up to 30% in all comparisons with the other methods and at the control point 45% compared to the RUFS and UDFS methods. At the control points 75% and 90%, although KSUFS is not the control method, none of the control algorithms according to the sum of ranks of the *Aligned Friedman* tests are able to obtain significant differences against it.

The previous accuracy results show that KSUFS out-stands over the rest of cluster-aware methods selecting features in the first set of control points (up to the control point 30-45%), as it occurs in the case of non-cluster-aware methods. From these percentages of selected features, methods tend to obtain similar performance results –there are even some algorithms that obtain a better position than KSUFS according to the *Aligned Friedman* test. However, in these cases, no statistical differences are found against KSUFS. Therefore, if KSUFS properly works when lower amounts of attributes are chosen to be selected and it obtains equivalent results to the rest of the feature selection methods when higher amounts of features are going to be selected, its general usage can be recommended for feature selection with standard data.

TABLE 8. Accuracy results of KSUFS versus not applying feature selection (None) with big data problems.

Metric	15%	30%	45%	60%	75%	90%	Best	None
ACC	59.07	**59.49**	**60.28**	**60.51**	**60.06**	**60.33**	**61.85**	59.18
$best_{20}$	**13**	12	**13**	**17**	**16**	**17**	**18**	✗
R^+	**101.5**	**105.5**	**148.5**	**171.5**	**147.5**	**174.0**	**181.5**	✗
R^-	88.5	84.5	61.5	18.5	42.5	36.0	8.5	✗
p_{Wil}	7.78E-01	6.58E-01	1.10E-01	**1.07E-03**	**3.42E-02**	**8.31E-03**	**1.11E-04**	✗

VII. ANALYSIS OF RESULTS WITH BIG DATA

This section focuses on the analysis of the results of the unsupervised feature selection methods and the version of KSUFS designed to work with big data (Section IV). Similar to the study with standard data, three comparisons are considered in this section: comparison of KSUFS and not preprocessing (Section VII-A), the non-cluster-aware methods (Section VII-B) and the cluster-aware methods (Section VII-C).

A. COMPARISON OF USING FEATURE SELECTION AND NOT PREPROCESSING WITH BIG DATA

Table 8 shows the performance of KSUFS versus not applying feature selection (None). The results of KSUFS that outperform those of None are highlighted in bold.

The accuracy results with big data show the good behavior of KSUFS versus None: they are better at each control point than those of None with the exception of the control point 15%, which is slightly worse. The number of datasets in which each method obtains the best results is always higher for KSUFS at all the control points. These results particularly out-stand for the intermediate-high control points (60-90%), in which KSUFS is the best in almost all the big data problems considered (16-17 datasets out of 20).

The results of *Wilcoxon*'s tests show that KSUFS obtains more ranks than None at each control point, being better as the control point increases. The associated p-values show that the differences found also follows this criterion and they are more representative as the control point increases – these are significant when more than 60% of the features are selected.

Considering the results of each dataset in the control point with the best performance for KSUFS (column *Best*), the results clearly shows the better performance of applying feature selection, since it obtains a higher performance than None in almost all the datasets (18 out of 20) and the differences found are significant (p_{Wil} =1.11E-04).

These results show the suitability of KSUFS working with big data that, in addition to have less computational complexity than the version to treat standard data, it also allows to improve the performance of not preprocessing in the datasets considered. The analysis of results shows that KSUFS out-stands with respect to None in the intermediate-high control points (upper 60% of features selected), in which *Wilcoxon*'s *p*-values shows significant differences. This conclusion is the opposite to that obtained for standard data, in which KSUFS particularly out-stands for low-intermediate control points. This fact may be explained because considering data with less features, a lower amount of them are necessary to check the differences among KSUFS and None, because these variables are likely to contain a larger amount of structural information of the dataset. However, when considering hundreds or thousands of features, since they are likely to contain a lower amount of structural information to describe the data, a larger amount of variables are necessary to show an advantage of KSUFS in performance. Finally, similar to the case of standard datasets, the results of the approach *Best* considering big data are also superior to those of None. Thus, the existence of a percentage of features to be selected providing a better performance for feature selection than not applying preprocessing is shown.

TABLE 9. Accuracy results of KSUFS against non-cluster-aware feature selection methods with big data.

Metric	Method	15%	30%	45%	60%	75%	90%
ACC	KSUFS	**59.07**	**59.49**	**60.28**	**60.51**	**60.06**	**60.33**
	LSFS	57.82	58.36	58.83	59.07	57.84	58.83
	MVFS	57.11	57.86	57.77	58.52	57.66	58.77
	SPEC	52.26	55.49	56.57	57.85	57.80	58.30
	SPFS	54.91	56.30	58.40	59.75	59.18	58.17
$best_{20}$	KSUFS	**8**	**9**	**12**	**9**	**10**	**13**
	LSFS	6	3	2	3	3	5
	MVFS	4	5	4	4	4	5
	SPEC	1	1	4	4	4	4
	SPFS	3	4	3	5	5	5
Ranks	KSUFS	**31.38**	**32.00**	**30.75**	**29.48**	**33.18**	**23.48**
	LSFS	43.18	44.93	47.35	57.45	58.03	53.05
	MVFS	47.20	49.73	55.48	57.88	58.10	51.78
	SPEC	70.90	66.55	68.40	61.75	60.18	64.78
	SPFS	59.85	59.30	50.53	45.95	43.03	59.43
p_{AF}	-	**2.06E-03**	**1.92E-03**	**2.04E-03**	**1.98E-03**	**2.09E-03**	**2.06E-03**
p_{Fin}	KSUFS	✗	✗	✗	✗	✗	✗
	LSFS	1.98E-01	1.59E-01	**7.04E-02**	**3.92E-03**	**1.31E-02**	**1.69E-03**
	MVFS	1.11E-01	**7.05E-02**	**1.40E-02**	**3.92E-03**	**1.31E-02**	**2.04E-03**
	SPEC	**6.60E-05**	**6.63E-04**	**1.62E-04**	**1.74E-03**	**1.29E-02**	**2.70E-05**
	SPFS	**3.82E-03**	**5.84E-03**	**4.13E-02**	**7.25E-02**	2.83E-01	**1.78E-04**

B. COMPARISON WITH NON-CLUSTER-AWARE METHODS USING BIG DATA PROBLEMS

The accuracy of the non-cluster-aware techniques for big data at each control point are shown in Table 9. This table presents the performance of each feature selection method (rows *ACC*), the number of datasets in which each method obtains the best result (rows $best_{20}$), the results of the *Aligned Friedman* test (rows *Ranks* and p_{AF}) and the *p*-values associated to each comparison with the *Finner* test (rows p_{Fin}).

Analyzing these results, the performance of KSUFS is better than that of the other non-cluster-aware methods at each control point. Two clear tendencies are appreciated for the behavior of the rest of the methods. Below the control point 45%, LSFS is the second best method, whereas SPEC and SPFS usually obtain lower performances. MVFS obtains intermediate performances compared to the rest of techniques. However, once the control point 45% is passed, SPFS starts to change its behavior, being the second best method for the highest control points. Regarding to the number of datasets with best results, KSUFS is the best at all the control points, obtaining the largest amounts of datasets with the best results at intermediate-high control points (upper 45%).

The statistical analysis using the *Aligned Friedman* test provides similar conclusions, being KSUFS the best method highly differentiated from the rest. The low *p*-values p_{AF} indicate that the *Finner* test can be applied to check the significance of these differences. Thus, the *Finner* test shows that the differences found among these algorithms are significant in favor of KSUFS against all the non-cluster aware methods considered. Some exceptions are observed at the lowest control points: no statistical differences are found with respect to LSFS and MVFS at 15% and the same occurs for LSFS at 30%. The better behavior of SPFS at the highest control points is reflected by the fact that no statistical differences are found in its comparison with KSUFS at 75%.

The analysis of the previous results shows that KSUFS is able to obtain the best result in many of the datasets and control points with respect to the rest of non-aware-cluster methods, especially when considering medium and high control points (those higher than a 45%). The good working of KSUFS is especially remarkable in the last control points. This fact is in accordance with those results of KSUFS versus None (Section VII-A). Contrary to the case of standard data, when higher percentages of features are selected in big data, there is still a large number of features to be selected in the last control points that may influence in the performance results. This fact also influences to find differences among feature selection methods, since larger amounts of features may be necessary and, thus, differences are more clearly appreciated at intermediate-high control points.

C. COMPARISON WITH CLUSTER-AWARE METHODS USING BIG DATA PROBLEMS

Table 10 shows the results of cluster-aware methods (MCFS, NDFS, RUFS and UDFS) versus KSUFS with big data. The performances show that the best method is KSUFS. MCFS and NDFS usually share the second and third position, depending on the control point, whereas RUFS and UDFS are generally in the last positions. These two methods (RUFS and UDFS) obtaining the worst results with big data are those usually placed in second and third place with standard data, whereas

TABLE 10. Accuracy results of KSUFS against cluster-aware feature selection methods with big data.

Metric	Method	15%	30%	45%	60%	75%	90%
ACC	KSUFS	**59.07**	**59.49**	**60.28**	**60.51**	**60.06**	**60.33**
	MCFS	58.49	57.36	58.44	57.81	58.00	57.85
	NDFS	57.02	58.35	57.56	58.19	58.08	58.06
	RUFS	57.21	57.29	57.58	58.03	57.38	58.51
	UDFS	54.55	55.83	57.09	58.11	57.72	57.63
$best_{20}$	KSUFS	**7**	6	**10**	**11**	**14**	**14**
	MCFS	5	2	4	4	7	3
	NDFS	6	7	4	3	2	5
	RUFS	6	**8**	4	3	3	5
	UDFS	3	2	3	2	5	4
Ranks	KSUFS	**36.93**	**28.73**	**27.00**	**28.80**	**26.95**	**24.13**
	MCFS	43.28	57.80	50.83	60.13	59.28	59.83
	NDFS	54.10	48.55	57.20	54.80	47.73	49.45
	RUFS	49.98	54.10	56.15	56.48	56.80	51.80
	UDFS	68.23	63.33	61.33	52.30	61.75	67.30
p_{AF}	-	**1.95E-03**	**1.88E-03**	**1.94E-03**	**1.86E-03**	**2.14E-03**	**2.10E-03**
p_{Fin}	KSUFS	✗	✗	✗	✗	✗	✗
	MCFS	4.89E-01	**3.06E-03**	**9.41E-03**	**2.55E-03**	**8.52E-04**	**1.99E-04**
	NDFS	1.19E-01	**3.07E-02**	**1.99E-03**	**6.12E-03**	**2.35E-02**	**5.77E-03**
	RUFS	2.01E-01	**7.56E-03**	**1.99E-03**	**5.11E-03**	**1.52E-03**	**3.41E-03**
	UDFS	**2.58E-03**	**6.49E-04**	**7.32E-04**	**1.04E-02**	**5.95E-04**	**1.00E-05**

the best methods with big data (MCFS and NDFS) are positioned in the worst positions with standard data.

Regarding the number of datasets, KSUFS provides the best result at all the control points, with the exception of 30% in which RUFS is the method with more datasets with the best result. At the first control points (15-30%), the best results are distributed among several methods (mainly KSUFS, RUFS and NDFS). However, from 45% onwards, KSUFS is clearly the method obtaining the best results in a larger number of datasets (14 out of 20 at the control points 75-90%).

The *Aligned Friedman* tests shows that KSUFS obtains the best position at all the control points. The p_{Fin} values show that there are significant differences in favor of KSUFS at all the comparisons performed, with the exceptions of MCFS, NDFS and RUFS at the control point 15%. This fact shows again that KSUFS works better with big data when higher amounts of features are selected.

From the aforementioned analysis of results, it can be appreciated that the performance of KSUFS using the big data problems studied are better than those of the other cluster-aware feature selection methods. Thus, even though in some isolated control points the results of KSUFS are not statistically better than those of other feature selection methods, it offers good performance results in most of the cases, even though it does not know the number of clusters in the datasets like the other cluster-aware methods.

VIII. LESSONS LEARNED

In this research a novel unsupervised feature selection method based on non-parametric statistical tests, denoted as KSUFS, has been proposed and a thorough empirical study has been performed to check its behavior against other well-known and representative unsupervised feature selection techniques considering both standard and big data. From the results shown in the previous sections and their corresponding analysis, several lessons can be learned:

1) **On the utility of applying feature selection in unsupervised problems.** The performance results of Tables 5 and 8 show that KSUFS is generally better than not applying preprocessing for some of the control points studied. Furthermore, these results also show that for almost all the datasets studied, there is at least one control point in which KSUFS provides a higher performance than not preprocessing. These facts clearly reflect the benefits in terms of performance of applying feature selection with respect to not preprocessing, which are added to the obvious reduction of memory requirements to store the dataset and the lower processing time of data analysis algorithms used later.
2) **On the connection between unsupervised feature selection and data dimensionality.** In data with lower amounts of features, each of them is more likely to contain a higher charge of structural information. On the contrary, larger amount of features may imply that the description of data conformations is distributed among many of them. Thus, it may be relevant for feature selection methods being exact selecting the first most influential features in standard data, since each of them may contain important information to describe the data, whereas more features may be necessary to appreciate the effect of feature selection with big data.
3) **On the working of unsupervised feature selection methods.** In non-cluster-aware methods, LSFS and MVFS are usually followed by SPEC and SPFS dealing with standard data. However, with big data, LSFS and SPFS are exchanged, outstanding SPFS for the highest control points. In cluster-aware methods with standard data, RUFS and UDFS are usually followed by MCFS and NDFS, but these two generally out-stand with big data. Note how methods working better and worse with standard data exchange positions when considering big data. Therefore, these method seems to be particularly adapted to deal with data of a specific dimensionality.
4) **On the working of KSUFS with standard and big data.** Tables 5 and 8 show that KSUFS outperforms not preprocessing with significant differences for the control point 30% with standard data and for the intermediate-high control points 60-90% with big data. On the other hand, the results in Tables 6-7 and Tables 9-10 show that KSUFS is generally better than the other non-cluster-aware and cluster-aware methods. With standard datasets, KSUFS particularly out-stands in the first control points (up to 30-45%), whereas with big data KSUFS out-stands when considering medium-high control points (higher than 30-45%). Thus, difference from the other methods

studied, KSUFS is able to highlight with both types of data dimensionality.

IX. CONCLUDING REMARKS

In this research, we have proposed a novel unsupervised feature selection method, known as KSUFS, based on the computation of estimated feature distributions and the usage of statistical tests to select the most representative features in an unsupervised dataset. An adaptation for KSUFS to work with big data has been also proposed. Our method has been compared to other well-known non-cluster-aware and cluster-aware unsupervised feature selection techniques in an experimental study that considers both standard and big data.

The results obtained show the benefits of applying our proposal with respect to not preprocessing. Furthermore, KSUFS is able to outperform the rest of non-cluster-aware and cluster-aware methods considered. With standard datasets, KSUFS works particularly well when feature selection involves low and intermediate percentages of variables to be chosen. Therefore, since KSUFS performs well when low-intermediate amounts of attributes are chosen to be selected and it obtains equivalent results to the rest of the methods when the highest amounts of features are selected, its usage can be recommended for feature selection with standard data. With big data, in which feature selection may show a higher impact, the behavior of KSUFS particularly highlights. It is usually better than other non-cluster-aware and cluster-aware methods at all the control points. However, difference from standard datasets, the advantages in favor of KSUFS are more noticeable when larger amounts of features are selected.

In future works we plan to study the influence of the k value when creating the estimated features in KSUFS, check the inclusion of other statistical tests and develop new unsupervised feature selection methods by incorporating evolutionary algorithms to select the most influential variables and automatically determine the optimal number features to be chosen.

ACKNOWLEDGMENT

José A. Sáez holds a *Juan de la Cierva-formación* fellowship (*Ref. FJCI-2015-25547*) from the Spanish Ministry of Economy, Industry and Competitiveness.

REFERENCES

[1] S. Arya and D. M. Mount, "Approximate range searching," *Comput. Geometry*, vol. 17, nos. 3–4, pp. 135–152, Dec. 2000.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *J. ACM*, vol. 45, pp. 891–923, Nov. 1998.

[3] G. E. A. P. A. Batista and M. C. Monard, "An analysis of four missing data treatment methods for supervised learning," *Appl. Artif. Intell.*, vol. 17, nos. 5–6, pp. 519–533, 2003.

[4] G. Bello-Orgaz, J. J. Jung, and D. Camacho, "Social big data: Recent achievements and new challenges," *Inf. Fusion*, vol. 28, pp. 45–59, Mar. 2016.

[5] M. Beyer and D. Laney, " The importance of 'big data': A definition," Gartner Res., Stamford, CT, USA, Tech. Rep. G00235055, 2012.

[6] V. Bolón-Canedo and A. Alonso-Betanzos, "Ensembles for feature selection: A review and future trends," *Inf. Fusion*, vol. 52, pp. 1–12, Dec. 2019.

[7] D. Chakraborty, V. Narayanan, and A. Ghosh, "Integration of deep feature extraction and ensemble learning for outlier detection," *Pattern Recognit.*, vol. 89, pp. 161–171, May 2019.

[8] R. Chen, N. Sun, X. Chen, M. Yang, and Q. Wu, "Supervised feature selection with a stratified feature weighting method," *IEEE Access*, vol. 6, pp. 15087–15098, 2018.

[9] D. Dua and E. K. Taniskidou. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

[10] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *J. Mach. Learn. Res.*, vol. 5, pp. 845–889, Jan. 2004.

[11] F. Honghai, C. Guoshun, Y. Cheng, Y. Bingru, and C. Yumei, "A SVM regression based approach to filling in missing values," in *Knowledge-Based Intelligent Information and Engineering Systems* (Lecture Notes in Computer Science), vol. 3683. Berlin, Germany: Springer, 2005.

[12] H. Finner, "On a monotonicity problem in step-down multiple test procedures," *J. Amer. Stat. Assoc.*, vol. 88, no. 423, pp. 920–923, 1993.

[13] J. W. Grzymala-Busse, L. K. Goodwin, W. J. Grzymala-Busse, and X. Zheng, "Handling missing attribute values in Preterm birth data sets," in *Proc. 10th Int. Conf. Rough Sets, Fuzzy Sets, Data Mining, Granular Comput.*, 2005, pp. 342–351.

[14] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, P. Schölkopf, and J. Platt, Eds. Cambridge, MA, USA: MIT Press, 2006, pp. 507–514.

[15] Z. He, M. Yang, Y. Gao, H.-D. Liu, and Y. Yin, "Joint multi-label classification and label correlations with missing labels and feature selection," *Knowl.-Based Syst.*, vol. 163, pp. 145–158, Jan. 2019.

[16] A. L'Heureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, "Machine learning with big data: Challenges and approaches," *IEEE Access*, vol. 5, pp. 7776–7797, 2017.

[17] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–45, Jan. 2018.

[18] J. Li and H. Liu, "Challenges of feature selection for big data analytics," *IEEE Intell. Syst.*, vol. 32, no. 2, pp. 9–15, Mar./Apr. 2017.

[19] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis," in *Proc. 26th Conf. Artif. Intell.*, 2012, pp. 1026–1032.

[20] J. Liu and E. Zio, "Integration of feature vector selection and support vector machine for classification of imbalanced data," *Appl. Soft Comput. J.*, vol. 75, pp. 702–711, Feb. 2019.

[21] L. Lovász and M. D. Plummer, *Matching Theory*. Amsterdam, The Netherlands: North Holland, 1986.

[22] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge Univ. Press, 2008.

[23] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*. Hoboken, NJ, USA: Wiley, 2004.

[24] J. Miao and L. Niu, "A survey on feature selection," *Procedia Comput. Sci.*, vol. 91, pp. 919–926, Jan. 2016.

[25] M. Osman, A. M. Abu-Mahfouz, and P. R. Page, "A survey on data imputation techniques: Water distribution system as a use case," *IEEE Access*, vol. 6, pp. 63279–63291, 2018.

[26] B. Parlak and A. Uysal, *On Feature Weighting and Selection for Medical Document Classification* (Studies in Computational Intelligence), vol. 718. Cham, Switzerland: Springer, 2018, pp. 269–282.

[27] F. Pourpanah, C. P. Lim, X. Wang, C. J. Tan, M. Seera, and Y. Shi, "A hybrid model of fuzzy min–max and brain storm optimization for feature selection and data classification," *Neurocomputing*, vol. 333, pp. 440–451, Mar. 2019.

[28] M. Qian and C. Zhai, "Robust unsupervised feature selection," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 1621–1627.

[29] J. Sáez, J. Derrac, J. Luengo, and F. Herrera, "Statistical computation of feature weighting schemes through data estimation for nearest neighbor classifiers," *Pattern Recognit.*, vol. 47, no. 12, pp. 3941–3948, 2014.

[30] J. Sáez, M. Galar, J. Luengo, and F. Herrera, "INFFC: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control," *Inf. Fusion*, vol. 27, pp. 19–32, Jan. 2016.

[31] J. Sáez, J. Luengo, J. Stefanowski, and F. Herrera, "SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering," *Inf. Sci.*, vol. 291, pp. 184–203, Jan. 2015.

[32] P. Singh, R. Sarkar, and M. Nasipuri, "Significance of non-parametric statistical tests for comparison of classifiers over multiple datasets," *Int J. Comput. Sci. Math.*, vol. 7, no. 5, pp. 410–442, 2016.
[33] N. V. Smirnov, "Estimate of deviation between empirical distribution functions in two independent samples," (in Russian), *Bull. Moscow Univ.*, vol. 2, no. 2, pp. 3–16, 1939.
[34] U. Stánczyk and L. C. Jain, Eds., *Feature Selection for Data and Pattern Recognition* (Studies in Computational Intelligence), vol. 584. Orlando, FL, USA: Springer, 2015.
[35] X. Tang, Y. Dai, and Y. Xiang, "Feature selection based on feature interactions with application to text categorization," *Expert Syst. Appl.*, vol. 120, pp. 207–216, Apr. 2019.
[36] C. Wan, Y. Wang, Y. Liu, J. Ji, and G. Feng, "Composite feature extraction and selection for text classification," *IEEE Access*, vol. 7, pp. 35208–35219, 2019.
[37] L. Wang and H. Shen, "Improved data streams classification with fast unsupervised feature selection," in *Proc. 17th Int. Conf. Parallel Distrib. Comput., Appl. Technol.*, Dec. 2016, pp. 221–226.
[38] S. Wang and H. Wang, "Unsupervised feature selection via low-rank approximation and structure learning," *Knowl.-Based Syst.*, vol. 124, pp. 70–79, May 2017.
[39] Y. Wang, Y. Yang, Y. Liu, and A. Bharath, "A recursive ensemble learning approach with noisy labels or unlabeled data," *IEEE Access*, vol. 7, pp. 36459–36470, 2019.
[40] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.
[41] X. Wu and V. Kumar, *The Top Ten Algorithms in Data Mining*, 1st ed. London, U.K.: Chapman & Hall, 2009.
[42] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "l_2, 1-norm regularized discriminative feature selection for unsupervised learning," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1589–1594.
[43] S. Yu, R. Liao, W. An, H. Chen, E. B. García, Y. Huang, and N. Poh, "GaitGANv2: Invariant gait feature extraction using generative adversarial networks," *Pattern Recognit.*, vol. 87, pp. 179–189, Mar. 2019.
[44] M. Yuan, Z. Yang, and G. Ji, "Partial maximum correlation information: A new feature selection method for microarray data classification," *Neurocomputing*, vol. 323, pp. 231–243, Jan. 2019.
[45] W. Yuan, D. Guan, T. Ma, and A. Khattak, "Classification with class noises through probabilistic sampling," *Inf. Fusion*, vol. 41, pp. 57–67, May 2018.
[46] Z. Zhao, L. Wang, H. Liu, and J. Ye, "On similarity preserving feature selection," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 3, pp. 619–632, Mar. 2013.
[47] W. Zhou, C. Wu, Y. Yi, and G. Luo, "Structure preserving non-negative feature self-representation for unsupervised feature selection," *IEEE Access*, vol. 5, pp. 8792–8803, 2017.

JOSÉ A. SÁEZ received the Ph.D. degree in computer science and computing technology from the University of Granada. He is currently working at the University of Salamanca, holding a "Juan de la Cierva-formación" Postdoctoral Grant from the Spanish Ministry of Economy, Industry and Competitiveness. His main research interests are related to data mining, data preprocessing, and data transformation tasks in knowledge discovery in databases, including noisy data in classification, discretization methods, imbalanced learning, performance evaluation methods, non-parametrical statistical tests, unsupervised learning, and others. His main research line is that of noisy data in classification tasks, counting with more than 20 publications on this topic.

EMILIO CORCHADO received the Ph.D. degree in computer science from the University of Salamanca.

He is currently a Full Professor in computer and automatic science with the University of Salamanca, Spain. His research interests include neural networks, with a particular focus on exploratory projection pursuit, maximum likelihood hebbian learning, self-organizing maps, multiple classifier systems, and Hybrid Systems. He has published over 100 peer-reviewed articles in a range of topics from knowledge management and risk analysis, intrusion detection systems, food industry, artificial vision, and modeling of industrial processes. He has been the Organizing Chair, the Program Committee Chair, the Session Chair and General Chair for a number of conferences, such as the International Conference on Hybrid Artificial Intelligence Systems (HAIS), the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), and the International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES). He is a RTD expert hired by international organizations such as the European Commission, the Grant agency of Czech Republic, the Spanish National Agency for Assessment and Forecasting, and has collaborated with SMEs and new companies in the innovation field in about 40 projects. He has patented software models and he owns the IP of more than 10 ICT tools and models. He was the Chair of the IEEE Spanish Section (2014–2015), and has actively contributed to several current projects in the EU, including SOFTCOMP, IT4Innovation, ICT Action COST IC1303, and IntelliCIS NISIS.