DOI 10.3233/IDA-160074
IOS Press

A novel data reduction method based on information theory and the Eclectic Genetic Algorithm

Edwin Aldana-Bobadilla[a,*], Ivan Lopez-Arevalo[b] and Alejandro Molina-Villegas[c]
[a]*CONACYT-CINVESTAV, Victoria, Tamaulipas, México*
[b]*CINVESTAV, Victoria, Tamaulipas, México*
[c]*CONABIO, Ciudad de México, México*

Abstract. A common task in data analysis is to find the appropriate data sample whose properties allow us to infer the parameters and behavior of the data population. In data mining this task makes sense since usually the population is significantly huge, and thus it is required (for practical reasons) to obtain a subset that preserves its properties. In this regard, statistics offers some sampling techniques usually based on asymptotic results from the Central Limit Theorem. The effectiveness of such ways is bounded by several considerations as the sampling strategy (simple with or without replacement, stratified, cluster-based, etc.), the size of the population and the dimensionality of the space of the data. Due to these considerations alternative proposals are necessary. We propose a method based on a measure of information in terms of Shannon's Entropy. Our idea is to find the optimal sample whose information is as similar as possible to the information of the population, subject to several constraints. Finding such sample represents a hard optimization problem whose feasible space disallows the use of traditional optimization techniques. To solve it, we resort to a breed of Genetic Algorithm called Eclectic Genetic Algorithm. We test our method with synthetic datasets; the results show that our method is suitable. For completeness, we used several datasets from real problems; the results confirm the effectiveness of our proposal and allow us to visualize different applications. Finally, we establish a baseline based on selection instance methods as a point reference to measure the effectiveness of our method.

Keywords: Data reduction, sampling data, instance selection, genetic algorithms, Shannon's Entropy

1. Introduction

A first approach to data reduction is the sampling. The goal of sampling is to choose a representative subset S from a set called population denoted by P. One way in which S may be obtained is by a random process where each element in P has an equal probability of being selected (simple sampling). When this process allows us to choose an element from P more than once, it is called sampling with replacement [1], otherwise it is called sampling without replacement [2]. Alternative ways to obtain S are: systematic sampling [3], stratified sampling and cluster sampling [4]. Regardless of the *sampling strategy*, an important concern is how to determine the cardinality or size of S. Usually this value is

[*] Corresponding author: Edwin Aldana-Bobadilla, CONACYT-CINVESTAV, Victoria, Tamaulipas, México. E-mail: ealdana@tamps.cinvestav.mx.

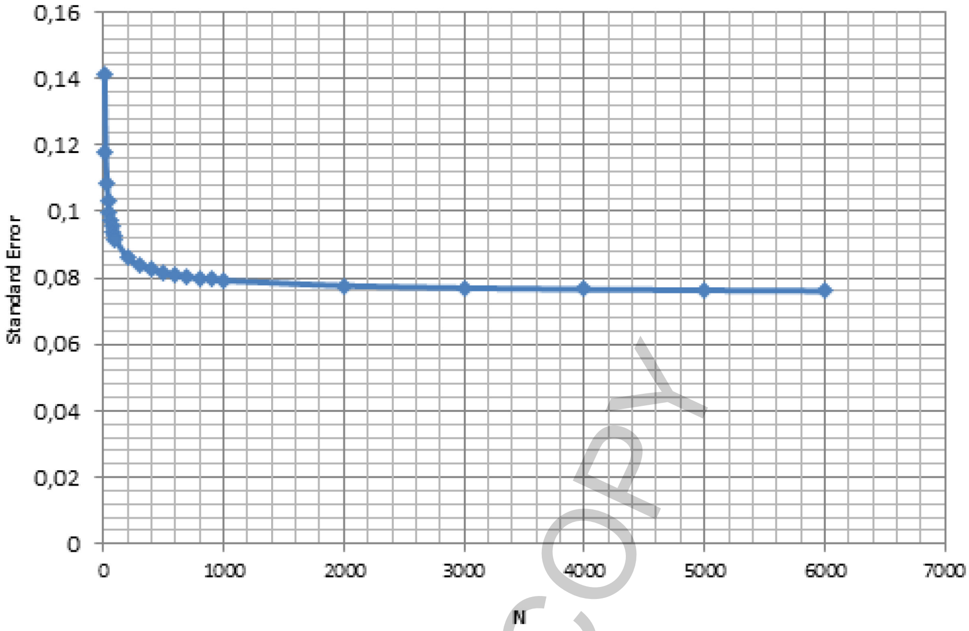


Fig. 1. Standard error in function of the sample size.

determined resorting to asymptotical results from the Central Limit Theorem (CLT) [5]. Assuming a sample S_i of size n drawn from P with mean μ_{S_i}, let $\bar{X}$ be a set of k means μ_{S_i} of the form:

$$\bar{X} = [\mu_{S_1}, \mu_{S_2}, ..., \mu_{S_k}] \tag{1}$$

From CLT, it is said that there is a relationship between the mean of $\bar{X}$ denoted as $\mu_{\bar{X}}$ and the mean of the population P denoted as μ, such relationship is given by:

$$\mu_{\bar{X}} \simeq \mu \tag{2}$$

Likewise, it is said that there is a relationship between the deviation of $\bar{X}$ denoted as $\sigma_{\bar{X}}$ and the standard deviation of the population σ, such relationship is given by:

$$\sigma_{\bar{X}} \simeq \begin{cases} \frac{\sigma}{\sqrt{n}}, & \text{sampling with replacement} \\ \frac{\sigma}{\sqrt{n}}\sqrt{\frac{N-n}{N-1}}, & \text{sampling without replacement} \end{cases} \tag{3}$$

where N is the cardinality of the set P. Since $\sigma_{\bar{X}}$ represents an error measure (usually called standard error) of the samples S_i, its value must be as small as possible. To satisfy this condition, an optimal value of n must be found. We illustrate this fact in Fig. 1 with a synthetic dataset in $\Re$ of size 6000. Every point is the standard error obtained with different values of n assuming that the sampling strategy is with replacement. We can see that the value of n that minimizes the standard error is the closest to the population size. Obviously, for practical purposes, such value is unsuitable.

From Eq. (3) the size of a sample (with replacement) can be written as:

$$n \simeq \frac{\sigma^2}{\sigma^2_{\bar{X}}} \tag{4}$$

Typically the value of $\sigma_{\bar{X}}$ is defined in a discretionary way and represents the permissible error of the sampling process. This fact implies the following remarks:

- Given the asymptotic nature of the Eqs (3) and (4), the value of n is not always appropriate. From our example (see Fig. 1), if we assume that $\sigma_{\bar{X}} = 0.05$, the value of n will be greater than the size or cardinality of P. So we are facing a problem which involves to find the break-even point between n and $\sigma_{\bar{X}}$.
- The result in Eq. (3) depends on the assumption that the elements from P are chosen by a simple random sampling with replacement. Another sampling strategy does not guarantee that such result is met.
- A generalization of the above asymptotic relationships to data in a multidimensional space may be computationally complex and even unfeasible in some cases.
- In many approaches of Data Mining (DM) and Machine Learning (ML) the task of finding the optimal sample goes beyond suggesting an adequately value of n in terms of CLT. This value should be based on the data properties rather than asymptotic assumptions.

1.1. A new approach

There are many ways of picking n elements from P. The optimal way is one in which the obtained sample S is as "similar" as possible to P in terms of those properties of P that are preserved by S. Specifically, we hypothesize that the optimal S is one sample of size n in which its information is as similar as possible to the information of P. To measure the information, we can resort to Shannon's Entropy [6] (see Subsection 3).

Given that $n \in 1, 2, ...N$, the problem of finding the **global optimal sample** S^* implies to explore a huge search space whose size is given by:

$$\sum_{n=1}^{N} \binom{N}{n} = \sum_{n=1}^{N} \frac{N!}{n!(N-n)!} = 2^N - 1 \qquad (5)$$

This expression represents the number of samples, each of size n, that can be formed from P (whose size is N). For example, given P with $N = 50$, there are 1125899906842620 different samples of size n. It gives us an idea of the huge feasible space of the problem.

For obvious reasons, we want that the value of n is as small as possible. However, the problem goes beyond finding the minimal value of n. A same value of n can define a wide set of ways (exactly $\binom{N}{n}$) to obtain subsets from P, some which might not meet conditions or desired properties relative to mentioned P. Specifically, in this work, we consider that an appropriate subset is one where both its size is minimum and its information (in term of Shannon's Entropy) relative to P is as similar as possible. This poses an optimization problem with a huge feasible space. It requires to resort to a method that allows us to explore such space efficiently.

Among the many methods that have arisen, we can mention tabu search [7], simulated annealing [8], ant colony optimization [9], particle swarm optimization [10] and evolutionary computation [11]. Furthermore, among the many variations of evolutionary computation, we find evolutionary strategies [12], evolutionary programming [13], genetic programming [14] and genetic algorithms (GAs) [15]. All of these methods are used to find approximate solutions for complex optimization problems. It was proven that an elitist GA always converges to the global optimum [16]. Such a convergence, however, is not bounded in time, and the selection of the GA variation with the best dynamic behavior is very convenient. In this regard, we rely on the conclusions of previous analyses, which showed that a breed of GA, called the Eclectic Genetic Algorithm (EGA), achieves the best relative performance [17,18]. In Appendix A, the interested reader can find important details about EGA.

Having determined a measure of information as criterion to find the most representative sample S^* drawn from P; and chosen the appropriate method to explore the wide search space, in the following sections, we present the details of our proposal, but not before presenting the related works.

1.2. Related works

The problem of finding an appropriate sample, can have two possible scenarios:

- One where the parameters and behavior of P are unknown and these must be inferred from a limited set of observations S.
- Another where there is a wide set of observations P, but due to practical reasons, it is required to obtain a subset S whose parameters and behavior are as similar as those of P.

The first scenario is common in those contexts in which the determination of the parameters of P may be impractical or unfeasible. For example, in many medical studies where the data involve patients with a disease or in those surveys about the political preferences of the inhabitants of a country. In these cases, usually the sampling is carried out based on the asymptotical results from the CLT. Since the parameters of P are unknown, the sampling may include several assumptions about them. Given the disadvantages of this approach, in particular contexts the need for finding other sampling ways has arisen. For instance, in [19] a method is presented for determining the appropriate sample size for a series of screening trials to identify promising new therapeutic agents. In [20] a work motivated by a specific problem in microarray experiments is shown; here the problem of choice of sample size is approached as part of a decision problem, involving both the sample size decision before carrying out the experiment and the later decision about the multiple comparisons once the data have been collected. In [21] a method that allows a relatively simple calculation of the required number of subjects in a reliability study is presented.

The second scenario is when given a large set P of observations about a phenomenon, it is required to obtain a subset S to infer its behavior model, since for practical reasons (often time or space) P cannot be used. This situation is common in Data Mining (DM) or Machine Learning (ML) approaches where the performance is compulsory. Often this problem is called *instance selection* [22]. In this regard several works have been published. Many of them have been focused on improving the performance of the Nearest Neighbor classifier (NN), since it is not suitable with very large datasets. Among these works, we mention Condensed NN (CNN) [23], Edited NN (ENN) and Repeated Edited NN (RENN) [24], Variable-kernel Similarity Metric (VSM) [25], Shrink and Growth [26]. These methods attempt to find iteratively an appropriate subset S which adequately classifies the remaining instances of P. Other approaches aim to remove instances systematically from P depending on the ability of the remaining instances to be well classified. This is the case of DROP [27] and the so-called Stratified Ordered Selection (SOS) [28]. Considering the instance selection as a hard search problem, some methods based on heuristics have arisen, for instance in [29] a method based on Random Mutation Hill Climbing RMHC is proposed, in [30] and [31] two approaches based on GAs are shown. Recently have arisen approaches based on the rough sets theory [32]. Among these approaches, we can mention the fuzzy-rough instance selection (FRIS) [33] and fuzzy-rough prototype selection method (FRPS) [34]. FRIS is based on the removal of instances that negatively affect the fuzzy positive region; instances are removed until there is no uncertainty among them. In FRPS the instances are ordered according to a measure based on fuzzy rough set theory to evaluate the lack of predictive ability of them. The instances for which the value exceeds a certain threshold are removed from the training set. The remain instances are high-quality instances to improve NN classification. We use the above methods to establish a baseline in order to compare the effectiveness of our proposal relative to it.

Clearly our proposal deals with the second scenario. In this regard, typically the cited methods have focused on finding the set S^* that optimizes the training process in classification tasks. Our proposal goes beyond, we want to find the optimal S^* regardless of the problem. It means that such method has potential applications both supervised and non-supervised problems, where the use of a limited dataset may be required. Like some cited methods, our proposal tackles the problem as an optimization problem. It differs from them in that the objective function is focus on minimizing the cardinality of the set S in such a way that, it preserves the information conveyed by P, instead of optimizing measures about the "classification ability" of S. As mentioned the measure of information is based on entropy.

To present our proposal, the rest of this work has been organized as follows: In Section 2 we show the background to guide the discussion about it. We show how to measure the information of the data based on Shannon's Entropy and how to extend such measure to data in a multidimensional space. In Section 3, we show important details as the objective function and the encoding to solve it through EGA. In Section 4, we show the experimental methodology and its results. Finally, we present the conclusions and infer several applications.

2. Background

In what follows, we provide the conceptual background which involves important details about how to measure the information of a dataset treated as a random variable. We introduce the entropy concept assuming an univariate random variable. Subsequently, we present a proposal to extend such concept to multivariate case.

2.1. Measuring the information of a dataset

The so-called entropy appeals to an evaluation of the information content of an univariate random variable Y with possible values y_1, y_2, y_r. From a statistical viewpoint, the information of the event $(Y = y_j)$ or simply $p(y_j)$ is inversely proportional to its likelihood. This information is denoted by $I(y_j)$, which can be expressed as:

$$I(y_j) = log\left(\frac{1}{p(y_j)}\right) \tag{6}$$

From information theory, the entropy of Y is the expected value of I. It is given by:

$$H(Y) = \sum_{i=1}^{r} p(y_j) log\left(\frac{1}{p(y_j)}\right) = -\sum_{i=1}^{r} p(y_j) log(p(y_j)) \tag{7}$$

Typically, the log function may be taken to be log_2, and then, the entropy is expressed in bits; otherwise, as $\ln$, in which case the entropy is in nats. We will use log_2 for the computations in this paper.

We can see that the entropy implies to determine the probabilities $p(y_j)$, for which, we need to know the probability distribution function (PDF) of Y. Since usually such PDF is unknown, a statistical inference approach is typically required. We propose to infer the PDF through a non-parametric approach, avoiding to make assumptions about a particular probability distribution (see Subsection 2.2). The term non-parametric does not imply absence of parameters; the idea is to keep the number of them as weak as possible. Non-parametric approaches can involve density functions, conditional density functions, regression functions or quantile functions to find the most suitable distribution [35].

So far, we have assumed that Y is an univariate random variable. However an important issue to be considered is the multivariate case. When $Y \in \mathbb{R}^d$, for $d \geqslant 2$, we have to determine $p(Y = \vec{y})$ that implies the joint probability $p(y_1, y_2, ...y_d)$. Evidently, when d increases such probability may be intractable from computational view point. In Subsection 2.2, we present a promising approach to tackle this problem.

Considering P and S as random variables, their expected values of information (in what follows simply information) can be calculated from Eq. (7). We want to choose a sample S_i of size n from P such that:

$$\frac{|H(S_i) - H(P)|}{H(P)} \leqslant \epsilon \tag{8}$$

where ϵ is a parameter that represents the maximum permissible error between the information of P and S_i. Since two different PDFs can have the same entropy, the Eq. (8) does not guarantee that S_i preserves the properties (at least from statistic view point) of P. For this reason we include additional constraints in order to assure that the distribution of S_i is as similar as possible to the distribution of P (see Subsection 3.1).

2.2. *Fitting distribution of Y*

Since determining the probability $p(y_{i1}, y_{i2}, ...y_{id})$ given a random variable $Y \in \mathbb{R}^d$ is imperative for our purposes, in this section, we present a method to approximate such probability. As first approach, we tackle the univariate case. Later we extend the ideas and results to multivariate case.

Given an univariate random variable Y (in what follows dataset), we can divide its space into a set of quantiles. A quantile q_i is an interval of the form $q_i = [\underline{y}, \overline{y}]$ where $\underline{y}$ and $\overline{y}$ are the lower and upper limit of q_i respectively. The quantile width denoted as $\triangle$ is given by:

$$\triangle = \frac{|max(Y) - min(Y)|}{m} \tag{9}$$

where m is a prior value of the desired number of quantiles (see Section 2.3). The first quantile is defined as a half-closed interval of the form:

$$q_1 = [min(Y), min(Y) + \triangle) \tag{10}$$

The subsequent quantiles can be defined as:

$$q_i = \begin{cases} [\overline{y}_{i-1}, \overline{y}_{i-1} + \triangle] \text{ if } i = m \\ [\overline{y}_{i-1}, \overline{y}_{i-1} + \triangle] \text{ otherwise} \end{cases} \tag{11}$$

where $\overline{y}_{i-1}$ is the upper limit of a previous quantile. In this case, $p(Y = y)$ can be approximated by the **proportion** of the elements that lie in the quantile to which y belongs. In Fig. 2 it illustrated a possible division of the space of Y into quantiles. Note that the number of elements that lie in a quantile determines its proportion, which can be an approximation to a probability value.

The above idea can be extended to higher dimensional data, in which case, a quantile will be a d-dimensional partition of the data space. In this way, given a dataset $Y \in \mathbb{R}^d$ with instances of the form $y = [y_1, y_2, ..., y_d]$, we can divide its space into a set of d-dimensional quantiles as it is illustrated in Fig. 3 for $d = 3$.

A d-dimensional quantile is composed by a set of intervals which determine the upper and lower limit for each dimension. Such definition is expressed as:

$$q_i = [[\underline{y}_{i1}, \overline{y}_{i1}], [\underline{y}_{i2}, \overline{y}_{i2}], ..., [\underline{y}_{id}, \overline{y}_{id}]] \tag{12}$$

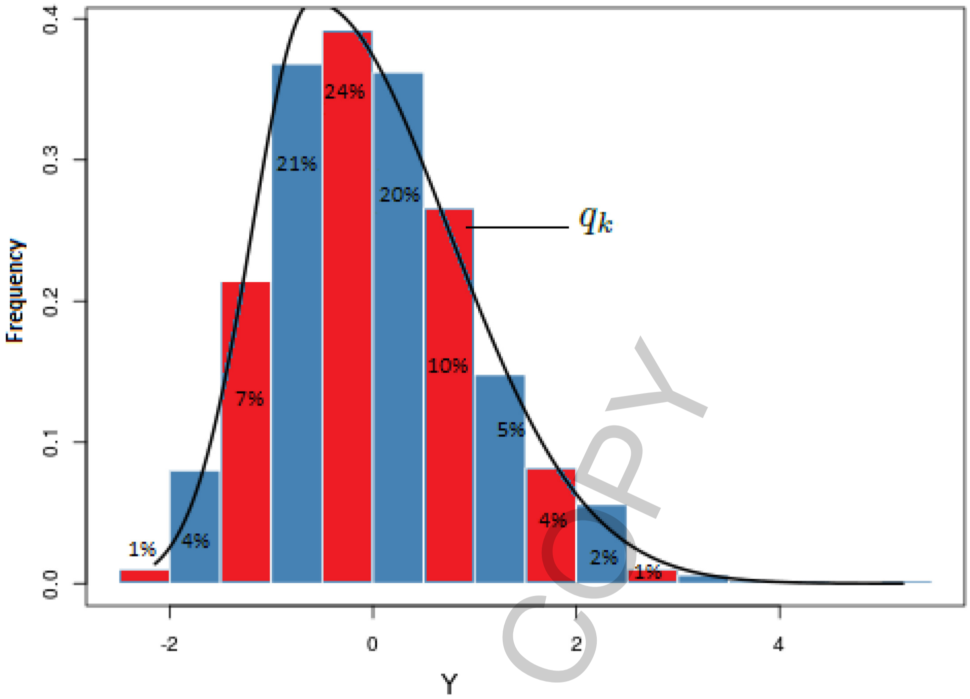


Fig. 2. Space discretization for a univariate variable.

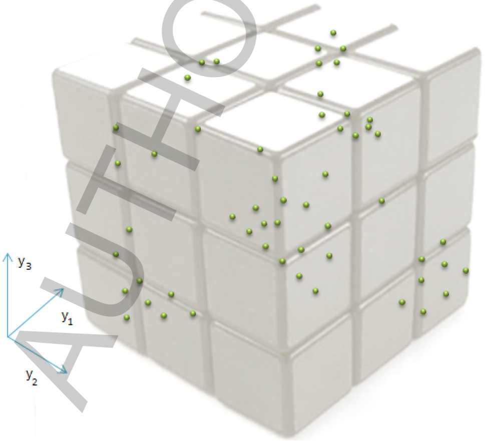


Fig. 3. Space discretization for a multivariate variable.

where $\underline{y}_{i,k}$ and $\overline{y}_{i,k}$ are the lower and upper limit of q_i in the k^{th} dimension. The width of each interval is given by:

$$\triangle_k = \frac{|max(Y_k) - min(Y_k)|}{m} \tag{13}$$

where Y_k is the dataset in the k^{th} dimension. Based on the above, we can generalize the way to determine the limits of a quantile when $Y \in \mathbb{R}^d$ as:

$$q_1 = \begin{bmatrix} [min(Y_1), min(Y_1) + \triangle_1) \\ [min(Y_2), min(Y_2) + \triangle_2) \\ ... \\ [min(Y_n), min(Y_d) + \triangle_n) \end{bmatrix}^T \tag{14}$$

for the first quantile, and:

$$q_i = \begin{cases} \begin{bmatrix} [\overline{y}_{(i-1),1}, \overline{y}_{(i-1),1} + \triangle_1] \\ [\overline{y}_{(i-1),2}, \overline{y}_{(i-1),2} + \triangle_2] \\ ... \\ [\overline{y}_{(i-1),d}, \overline{y}_{(i-1),1} + \triangle_n] \end{bmatrix}^T & \text{if } i = m \\ \\ \begin{bmatrix} [\overline{y}_{(i-1),1}, \overline{y}_{(i-1),1} + \triangle_1) \\ [\overline{y}_{(i-1),2}, \overline{y}_{(i-1),2} + \triangle_2) \\ ... \\ [\overline{y}_{(i-1),d}, \overline{y}_{(i-1),1} + \triangle_n) \end{bmatrix}^T & \text{otherwise} \end{cases} \tag{15}$$

for subsequent quantiles, where $\overline{y}_{(i-1),k}$ is the upper limit of a previous quantile $(i - 1)$ in the k^{th} dimension.

As univariate case, the PDF of Y is approximated by the proportion of the elements that lies in each quantile. In general given a random variable of the form $Y = [y_1, y_2, ..., y_d]$ the probability $p(y_1, y_2, ..., y_d)$ is the density (in terms of the proportion) of the quantile q_i to which the vector $[y_1, y_2, ..., y_d]$ belongs. Based on the above, now we can approximate the PDF of P and S in order to determine their entropy.

2.3. Determining the number of quantiles

To determine the value of m (number of quantiles), typically, Sturges' rule [36] and [37] is used. There are other alternative rules which attempt to improve the performance of Sturges's rule without a normality assumption as Doane's formula [38] and the Rice rule [39]. In this paper, we prefer the Rice rule, which is to set the number of intervals to twice the cube root of the number of instances or observations. In the case of 1000 instances, the Rice rule yields 20 intervals instead of the 11 recommended by Sturges' rule.

3. Proposal

Having defined the way to measure the information conveyed by P and S, in the following subsections, we present important details of our proposal.

3.1. Defining the objective function

We want to find the minimal value of n that allows us to obtain a sample S_i of size n drawn from P. In this regard, the objective function can be defined as:

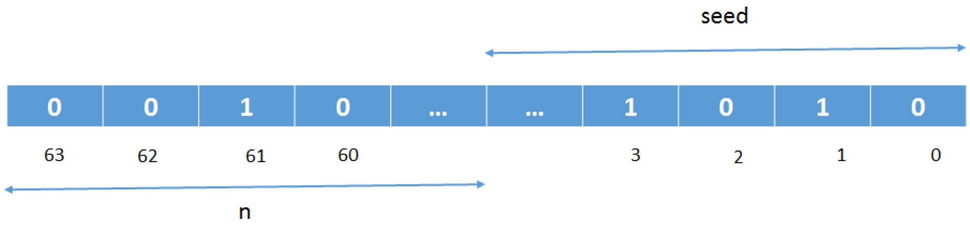


Fig. 4. Encoding of the problem.

$$\text{Minimize:} f(n) = \frac{n}{N} \tag{16}$$

The value of the objective function tends to 1 when the value of n is close to N (recall that N is the size of P). Evidently such function is not enough, the sample S_i (of size n) must also satisfy the following constraints:

1. The error between the information of P and S_i must be less than or equals to ϵ (see Eq. (8)).
2. As mentioned, the quantiles are partitions of the space of a dataset. Since S_i and P are datasets such that $S_i \subseteq P$, the set of quantiles can be unique to both P and S_i. We want that given P and S_i **the proportion** of elements of P that lie in the quantile q_k is as similar as possible to the proportion of elements of S_i that lie in this same quantile.

Thus, the objective function is given by:

$$\begin{array}{c} \text{Minimize: } f(n) = \frac{n}{N} \\ \\ \text{subject to:} \\ \frac{|H(S_i)-H(P)|}{H(P)} \leqslant \epsilon \\ \frac{1}{m}\sum |q_k(S_i) - q_k(P)| \leqslant \delta \\ n \in [2, N) \end{array} \tag{17}$$

where $q_k(S)$ and $q_k(P)$ are the proportion of elements of S and P that belong to k^{th} quantile respectively. We define the difference between these proportions as an error measure and establish that its average value must be less than or equals to δ. In this work we set $\epsilon = 0.05$ and $\delta = 0.01$, these values allow us to obtain a fast convergence to the optimal solution of Eq. (17).

3.2. *Encoding the objective function*

EGA proposes ω candidate solutions to obtain the best ω samples S_i drawn from P. We encode a candidate as a binary string of length 64. The 32 most significant bits encode an unsigned integer that corresponds to the value of n. The subsequent bits encode a unsigned integer number that corresponds to the random seed with which the sample S_i is chosen from P. In Fig. 4 this encoding is illustrated.

For each candidate a fitness value is calculated based on the objective function (see Eq. (17)). The fitness of those candidates that do not satisfy the constraints is punished through a penalty function (see Subsection 3.3). Evolution takes place after the repeated application of genetic operators of EGA. When the maximum number of iterations is reached, we have ω candidates which encode the best values (n and a random seed) to obtain S_i. The optimal solution will be the top 1 candidate. Important details about this process are shown in Subsection 3.4.

3.3. Constraints handling strategy

We are facing a constrained optimization problem, to solve it, the most common way is resorting to a penalty function [40]. In this approach, the objective function in Eq. (17) can be transformed as follows:

$$f(n) = \begin{cases} f(n) & \text{if } n \text{ is feasible solution} \\ f(n) + \textit{penalty(n)} & \text{otherwise} \end{cases} \tag{18}$$

There are many variations of penalty functions. Based on the results of a comprehensive analysis reported in [41], we use the method that exhibited the best performance, where the objective function $f(n)$ is handled as follows:

$$f(n) = \begin{cases} [K - \sum_{i=1}^{s} \frac{K}{p}] & \text{if } s \neq p \\ f(n) & \text{otherwise} \end{cases} \tag{19}$$

where K is a large constant $[O(10^9)]$, p is the number of constraints and s is the number of these which have been satisfied. The value of K induces a strictly separation of those individuals that satisfy $1, 2, \ldots, p$ constraints. This separation allows favoring those solutions that satisfy the largest number of such constraints.

3.4. Searching the optimal sample

When the evolutionary process of EGA is finished, the set of candidate solutions contains the fittest candidate whose genome encodes the optimal size and a random seed denoted as n^* and r respectively. It allows us to find the optimal sample S^* of size n^* drawn from P. In what follows, we describe this process:

EGA starts with a random set C of candidate solutions in accordance to the problem encoding. The cardinality of C is denoted as Θ. For each candidate $c_i \in C$, its genome is decoded to obtain a value of n and r. With these values, a random sample S_i of size n from P is obtained using as random seed r. Given S_i, the fitness value of c_i is determined based on objective function. If required the fitness values of c_i is penalized (see Eq. (19)). Subsequently, the set C is sorted in ascending order, based on fitness values.

The evolutionary process of EGA is based on the premise of elitism. It means that the best candidate solutions of every iteration must be preserved. Based on the above, in each iteration, EGA duplicates the set C in order to apply the genetic operators (crossover and mutation) to a copy of C. The duplicate C (now with size 2Θ) is evaluated and sorted in ascending order, based on the fitness values of each c_i. Later, the worst Θ individuals are removed to obtain a set C with the Θ fittest candidates. The evolutionary process is repeated until convergence criteria are met (usually a given number of iterations).

As result of the above process, the top 1 candidate is selected from C. It contains the best optimal way (given by n and a random seed) to obtain S^*. The described process is shown in Algorithm 3. A "generic version" of EGA (independent of the problem) can be found in Appendix A.

3.5. Setting parameters

As mentioned ϵ is the maximum permissible error between the information of S relative to P. Meanwhile δ is the average error between the proportions of the quantiles given S relative to the proportions of these quantiles given P. Since they represent the upper bound of error measures, we would like that their value is as small as possible. For this reason, we decided to set $\epsilon = 0.05$ and $\delta = 0.01$. It means a ratio of loss information less than 5% and a ratio of "discrepancy" between the PDF of P and S less than 1%.

Algorithm 1: Searching the Optimal Sample

```
Data:
$\Theta$ = Number of candidate solutions
$C = \emptyset$, set of candidate solutions
$c_i = 0$, the $i^{th}$ candidate
$\vec{f} = \emptyset$, fitness array of the candidates solutions
Result: The top 1 candidate which encodes the best optimal way to obtain $S^*$
Generate a random set $C$ of candidate solutions.
$C$ = initialization($\Theta$);
Determine the fitness value of $c_i$ based on Eq. (17):
$f$ = evaluate($C$);
Sort candidates from best to worst based on their fitness:
sort($C$, $f$);
while convergence criteria are not met do
    duplicate($C$);
    $bottom = 2 * \Theta$;
    for $i = \Theta$ to $2 * \Theta$ do
        Generate a random number $R$;
        if $R > p_c$ then
            crossover($c_i$, $c_{bottom}$));
        end
        $bottom = bottom$-1
    end
    Mutate the population in $b2m$ randomly selected bits:
    mutate($C$);
    $f$ = evaluate($C$);
    sort($C$, $f$);
    Eliminate the worst $\Theta$ individuals from $C$
    $C$ = remove($C$);
    Return top($C$)
end
```

The parameters associated to EGA are shown in Table 1. The value of them was determined experimentally in a previous study (see [17]). This study showed that from statistical view point, EGA converges to optimal solution around such values when the problems are demanding (those with non-convex and multi-modal functions). Since the problem to be solved by EGA in this work lies in this category, we consider such values appropriate. We found that they allow us to obtain good results regardless of the dataset from which we want to obtain the optimal sample.

4. Results

We wanted to show some preliminary results as a first approach to the effectiveness of our method (see Subsection 4.1). Subsequently, we statistically evaluated such effectiveness in two ways: a) the ratio of data reduction (size reduction percentage of S^* relative to P) and b) the sampling error (difference of the statistical properties of S^* relative to P) (see Subsection 4.2).

Table 1
Parameters of evolutionary process

Parameter	Description	Value
P_c	Crossover probability	0.90
P_m	Mutation probability	0.01
Θ	Number of candidates	80
G	Number of iterations	100

Often the datasets have hidden information which are not possible to be characterized through statistical measures. We also wanted to guarantee that the subset S^* preserves this information. We attempted to measure the hidden information through the *classification ability* of S^* relative to P. In this regard, we used a suite of datasets that represent classification problems where the *class label* for each instance is known. Usually these sets are called *labeled dataset*. Our hypothesis is that given a large labeled dataset P_l, we can find an optimal subset S_l^* to train a classifier instead of using P_l. The effectiveness in this case is given by the percentage matching between the class labels found by a classifier trained with S_l^* and the class labels found by the same classifier trained with P_l (see Subsection 4.3).

As mentioned in Subsection 1.2, recently have arisen approaches based on the rough sets theory as fuzzy-rough prototype selection method (FRPS) and fuzzy-rough instance selection (FRIS) which reduce the number of instances or elements that must be selected from a dataset. For completeness, we use these methods to establish a benchmark baseline in order to compare the effectiveness of our proposal. The experimental process and results are described in Subsection 4.4.

4.1. Preliminary results

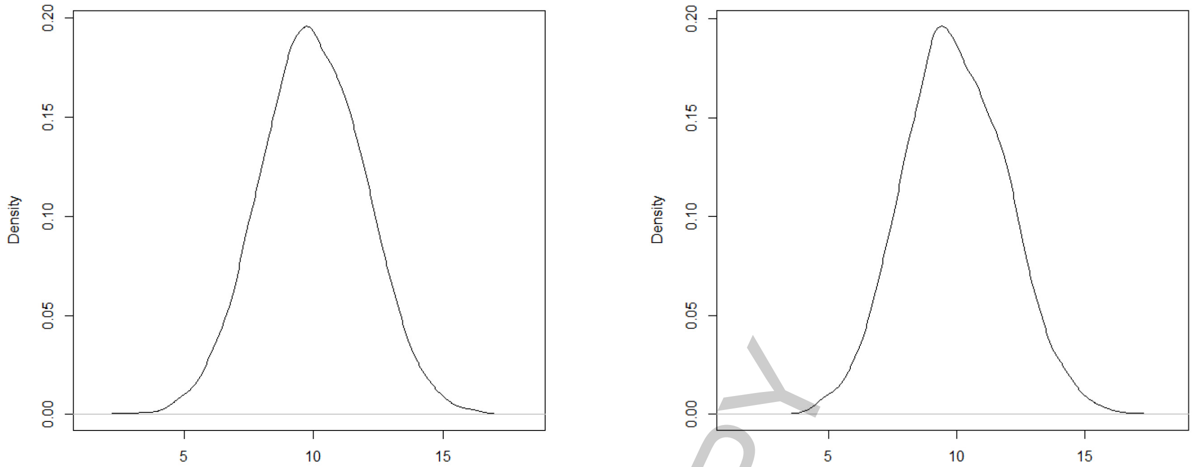
We executed preliminary experiments whose results allowed us to show that our method is promissory. The datasets in such experiments are described as follows:

- Dataset D_1 of 10000 elements in a one-dimensional space drawn from a *Gaussian distribution* with parameters $\mu = 10$ and $\sigma = 2$.
- Dataset D_2 of 10000 elements in a one-dimensional space drawn from a *Poisson distribution* with parameter $\lambda = 3$.
- Dataset D_3 of 10000 elements in a one-dimensional space drawn from a *Weibull distribution* with parameter $\lambda = 1$ (scale) and $k = 1.5$ (shape).
- Dataset D_4 of 10000 elements in a bi-dimensional space drawn from a *Gaussian distribution* with parameters $\vec{\mu} = [0.5, 0.5]$ and $\vec{\sigma} = [1.0, 1.0]$.
- Dataset D_5 of 100000 elements in bi-dimensional space that represent a *sinusoidal function* in the into the interval $[-2\pi, 2\pi]$.

Our hypothesis is that our method will find the optimal sample S^* of these datasets, retaining their statistical properties given by the PDF. In what follows some evidences are shown. In Figs 5, 6 and 7 we can see that the PDF of the sample is similar to the PDF of the original dataset. Such PDF retains some properties as unimodality, skewness and kurtosis.

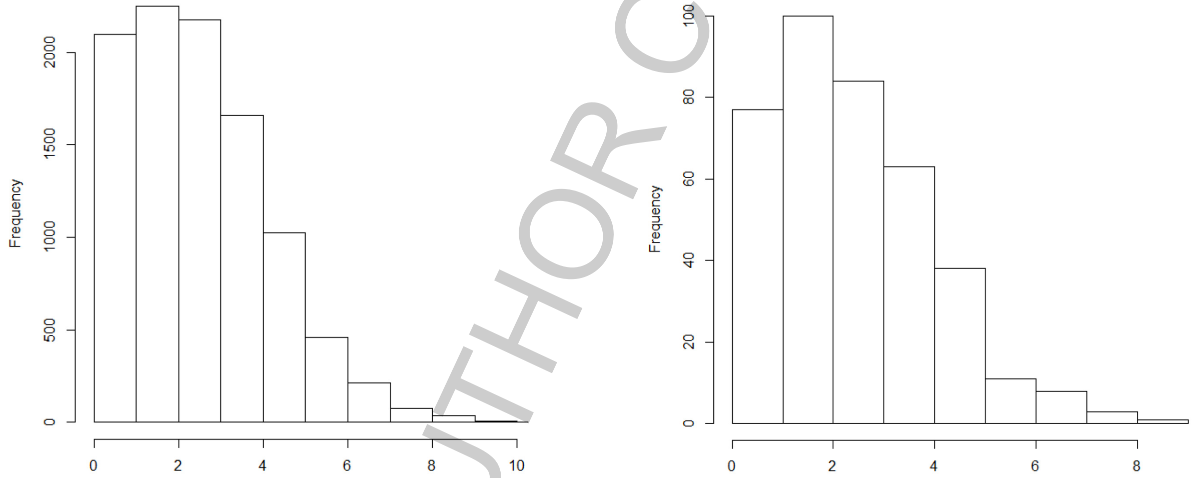
In the case of the **bi-dimensional** dataset D_4, the method found a sample S_4^* with the following parameters: $\vec{\mu} = [0.51, 0.48]$ and $\vec{\sigma} = [0.98, 0.96]$. We can see that such values are similar to those of the original data ($\vec{\mu} = [0.5, 0.5]$ and $\vec{\sigma} = [1.0, 1.0]$). For completeness in Fig. 8 the density plots of D_4 and S_4^* are shown.

Since the distribution of the **sinusoidal** dataset (D_5) is not characterized by a theoretical PDF, we decided to estimate its marginal distributions, and then we compared them against marginal distributions



(a) Density of D_1 (b) Density of S_1^* drawn from D_1 (n* = 144)

Fig. 5. Distribution of the dataset D_1 and the optimal sample S_1^*.



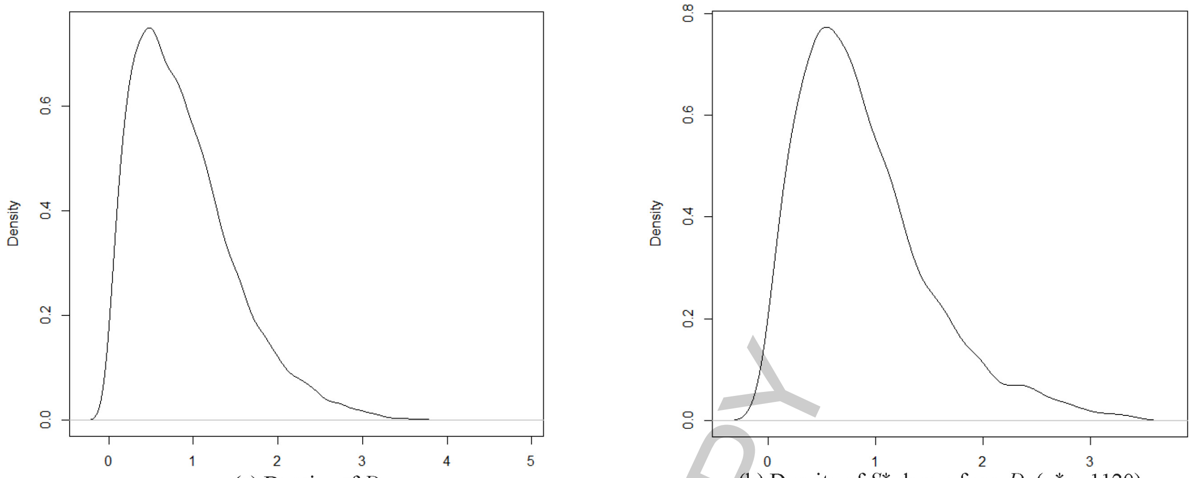
(a) Histogram of D_2 (b) Histogram of S_2^* drawn from D_2 (n* = 385)

Fig. 6. Distribution of the dataset D_2 and the optimal sample S_2^*.

of S_5^*. In Figs 9 and 10 the marginal distributions of D_5 and S_5^* in the abscissa (x) and ordinate (y) are shown.

In the case of y the similarity between distributions is evident. However, the distributions in x show a slight difference, for this reason, we wanted to analyze the data estimating the joint distribution, as it is shown in Fig. 11. We can see that the properties of the distribution are preserved by the sample in spite of there are marginal differences.

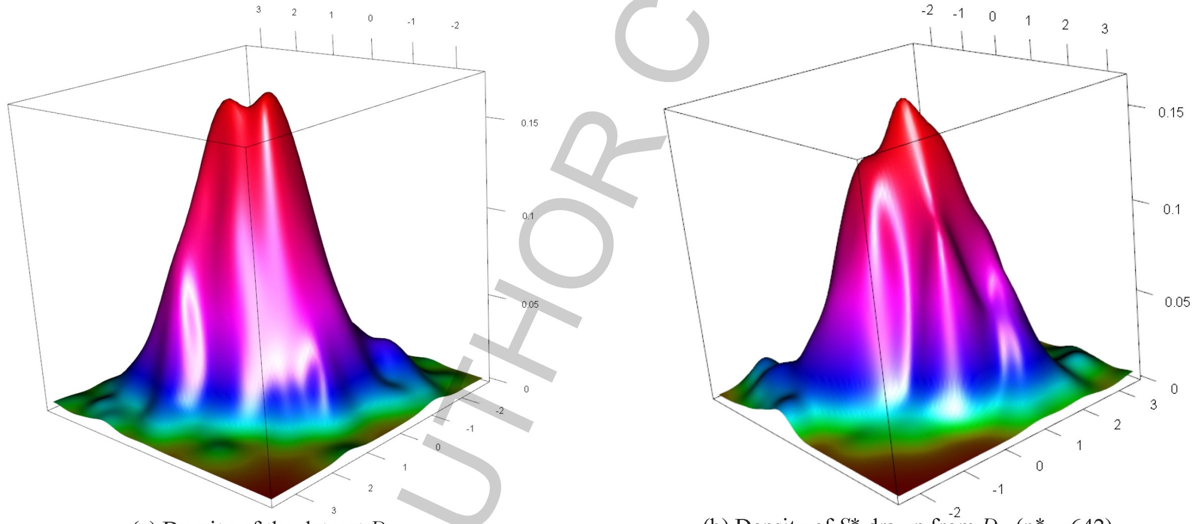
The above results show that the samples achieve to retain the information conveyed by the original data and reduce significantly its size. However, these results are not enough to generalize this observation. In the following subsection, we present an experimental methodology that allows us to generalize the effectiveness of our proposal.



(a) Density of D_3

(b) Density of S_3^* drawn from D_3 (n* = 1120)

Fig. 7. Distribution of the dataset D_3 and the optimal sample S_3^*.



(a) Density of the dataset D_4

(b) Density of S_4^* drawn from D_4 (n* = 642)

Fig. 8. Distribution of the dataset D_4 and S_4^*.

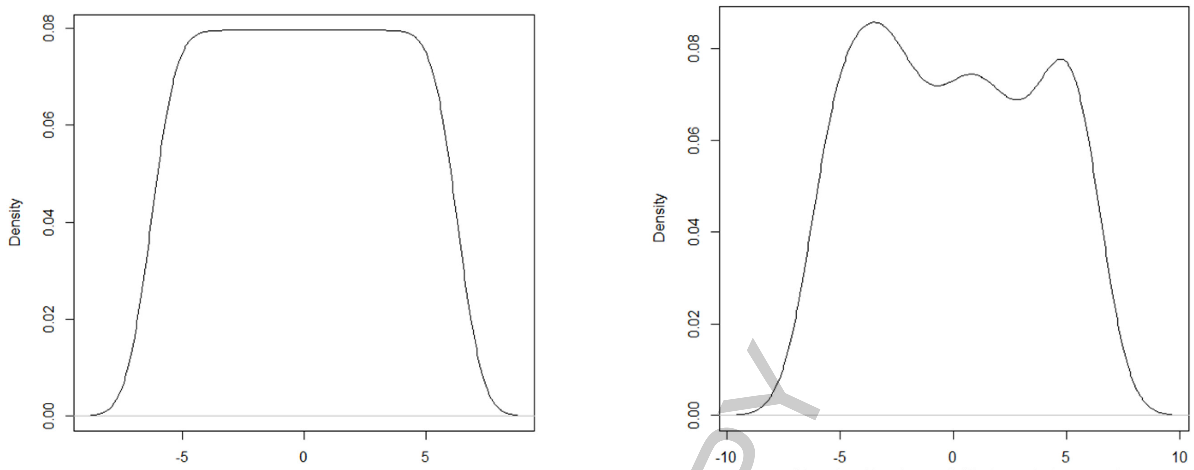
4.2. Statistical performance

We evaluated the effectiveness in two ways: a) the ratio of data reduction and b) the sampling error.

In the first case, we resort to a measure from data compression: the so-called *space saving* metric (SS) [42] which is a measure relative to the ratio between the size of a sample S_i and P, given by:

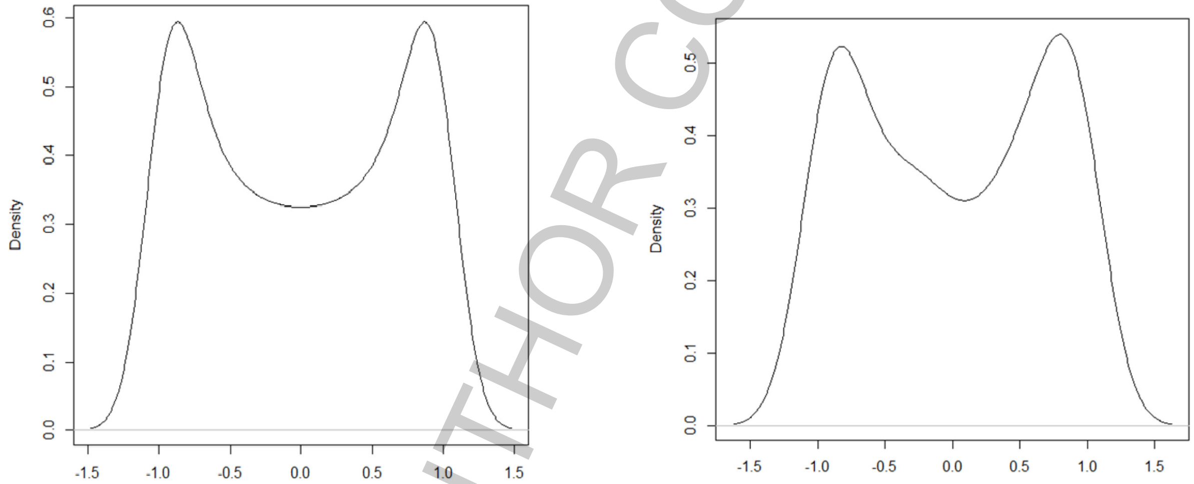
$$SS = 1 - \frac{n}{N} \tag{20}$$

A large value of SS (closer to 1) implies better performance. We calculated such metric with a wide set of experiments (about 5000) which included random datasets of size 1000, 5000, 10000 and 100000.



(a) Distribution of D_5 in x (b) Distribution of S_5^* in x (n* = 642)

Fig. 9. Marginal distribution of D_5 and S_5^* in x.



(a) Distribution of D_5 in y (b) Distribution of S_5^* in y (n* = 642)

Fig. 10. Marginal distribution of D_5 and S_5^* in y.

The average result is shown in Table 2. For completeness, we show the confidence interval of the results with a ρ-value of 0.05.

The experiments show that in average the sampling process allows us to reduce the size of the dataset in more than 70%.

Secondly, we want also to show that the "reduced dataset" (S^*) preserves the statistical properties of the original dataset (P). Usually such properties may be characterized by two statistics: μ and σ. We rely on them to define the following performance measures:

$$\begin{aligned} error_\mu &= \|\vec{\mu}_{S^*} - \vec{\mu}_P\| \\ error_\sigma &= \|\vec{\sigma}_{S^*} - \vec{\sigma}_P\| \end{aligned} \tag{21}$$

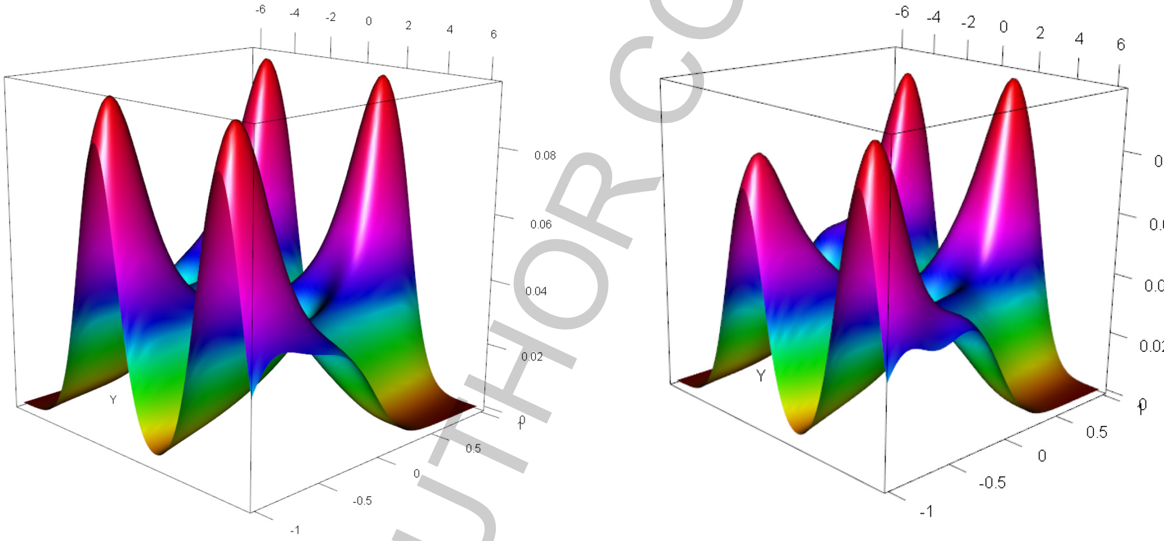
where $\vec{\mu}_{S^*}$, $\vec{\sigma}_{S^*}$ and $\vec{\mu}_P$, $\vec{\sigma}_P$ are the mean vector and the standard deviation vector of S^* and P respectively. The terms $error_\mu$ and $error_\sigma$ represent an Euclidean norm. Since there may be datasets

Table 2
Data reduction effectiveness

	SS
Average	0.7895
Standard deviation	0.3181
Lower limit	0.7806
Upper limit	0.7983
Confidence level	95%

Table 3
Sampling error characterized by $error_\mu$ and $error_\sigma$

	$error_\mu$	$error_\sigma$
Average	0.016	0.021
Standard deviation	0.00	0.031
Lower limit	0.00	0.031
Upper limit	0.00	0.031
Confidence level	95%	95%



(a) Joint distribution of D_5

(b) Joint distribution of S_5^* (n* = 642)

Fig. 11. Estimated joint distribution of D_5 and S_5^* in y.

with different scales, the norms achieved with them may be non-comparable. To avoid this problem, we scaled the data in P and S^* between 0 and 1. Thus, a small value of $error_\mu$ and $error_\sigma$ (closer to 0) implies better performance. During the execution of the experiments to obtain the average value of SS, the values of $error_\mu$ and $error_\sigma$ were also calculated which are shown in Table 3.

These results show that in general S^* retains about 98% of the statistical properties of P (characterized by μ and σ).

So far, we have shown that our method achieves to reduce the space of the dataset in more than 70% and preserves its statistical properties. However, typically the datasets have hidden information whose recognition through statistical measures is hard. We want to guarantee that the subset S^* also preserves this information. We discuss about this in the following subsection.

Table 4
Properties of the selected datasets

Dataset's name	Variables	Classes	Size	Missing values
Abalon	8	29	4,177	no
Cars	22	4	1,728	no
Census Income	14	2	32,561	yes
Hepatitis	20	2	155	yes
Yeast	10	8	1486	no

Table 5
Properties of the selected datasets

Dataset's name	Matching ratio	SS
Abalon	0.88	0.63
Cars	0.89	0.71
Census Income	0.94	0.76
Hepatitis	0.87	0.61
Yeast	0.92	0.71
Average	0.90	0.68

Table 6
Sampling error characterized by $error_\mu$ and $error_\sigma$

Method	SS	$error_\mu$	$error_\sigma$
Proposal	0.790	0.016	0.021
FRPS	0.937	0.247	0.238
FRIS	0.884	0.141	0.138

4.3. Preserving hidden information

In order to illustrate the effectiveness of our method to retain hidden information of a dataset when this is sampled, we selected 5 datasets (Abalone [43], Cars [44], Census Income [45], Hepatitis [46] and Yeast [47]) from the UCI Machine Learning repository whose properties are shown in Table 4. We chose labeled datasets that represent classification problems. Selection criteria of these datasets was based on the following features:

- Multi-dimensionality.
- Cardinality.
- Complexity (non-linearly separable problems).
- Categorical data.
- Data with missing values.

Some of these features involve pre-processing tasks to guarantee the quality of a dataset. We applied the following pre-processing techniques:

- Categorical variables were encoded using dummy binary variables [48].
- The datasets were scaled into $[0, 1)$.
- To complete missing information, we interpolate the unknown values with natural splines (known to minimize the curvature of the approximant) [49].

With these sets and resorting to a Bayesian Classifier (BC), we calculate the effectiveness as follows: Given a labeled dataset P_l, a subset without labels is obtained. This set is denoted as P_{test} which is used to assess the strength and utility of the predictive relationship determined by BC. Done this, an optimal sample S_l^* is obtained from P_l using our method. Based on this sample the training process of

BC is performed. Later, the predictive process of BC is executed with P_{test} in order to obtain a label vector $\vec{Y_1}$. Subsequently, the training process of BC is performed again, but this time with P_l. Then, the predictive process of BC is executed with P_{test} in order to obtain another label vector $\vec{Y_2}$. Finally, based on $\vec{Y_1}$ and $\vec{Y_2}$ a matching ratio is determined. This ratio allows us to measure the effectiveness of the S_l^* to train a classifier, relative to the entire dataset P_l. The described process is shown in Algorithm 2.

Algorithm 2: Matching ratio as effectiveness to classification

```
Data:
$P_l$ = Labeled dataset
$P_{test}$ = Test subset from $P_l$
$S_l^*$ = Subset from $P_l$ obtained by our method.
Result: Matching ratio
 Execute training process of BC with ($S_l^*$):
bayesianClassifier($S_l^*$);
Execute predictive process of BC with $P_{test}$:
$\vec{Y_1}$ = predict($S_l^*$);
Execute training process of BC with ($P_l$):
bayesianClassifier($P_l$);
Execute predictive process of BC with $P_{test}$:
$\vec{Y_2}$ = predict($S_l^*$);
Obtain matching ratio between $\vec{Y_1}$ and $\vec{Y_2}$:
$ratio$ = ratio($\vec{Y_1}$,$\vec{Y_2}$);
Return ($ratio$)
```

Based on the above, we determined the effectiveness for each dataset in Table 4. The results are shown in Table 5. For completeness we have included the index of data reduction SS.

These results show that beside reducing the size of the data, our method also preserves hidden information (one that cannot be characterized statistically) which is necessary to the classification process. We can see that the effectiveness (given by matching ratio) is about 90%. It means that our method achieved to obtain samples that allowed us to train a classifier as well as the original datasets. Also, it was possible to reduce the size of such datasets by 68% (in average).

4.4. Benchmarking

As a first approach, we use a synthetic Gaussian dataset in $\Re^2$ with $N = 10000$. In Fig. 12 are shown the results obtained after executing FRPS given this dataset. We can see that this method achieves a significant reduction of the number of instances. However, the marginal PDF of the remaining instances does not correspond to the marginal PDF of the dataset. It means that the obtained sample does not preserve the statistical properties of the original dataset.

Figure 13 shows the results obtained after executing FRIS. We can see a improvement, since the method achieves to obtain a sample (larger than the obtained by FRPS) that preserves in some degree the statistical properties of the dataset given its marginal PDFs.

In Fig. 14 we can see the results obtained by our proposal. We can see that it achieves a sample whose marginal PDFs are more similar to marginal PDFs of the dataset, in contrast with the samples obtained by FRPS and FRIS. It should be noted that the size of this sample is between 378 and 1885 (sizes sample

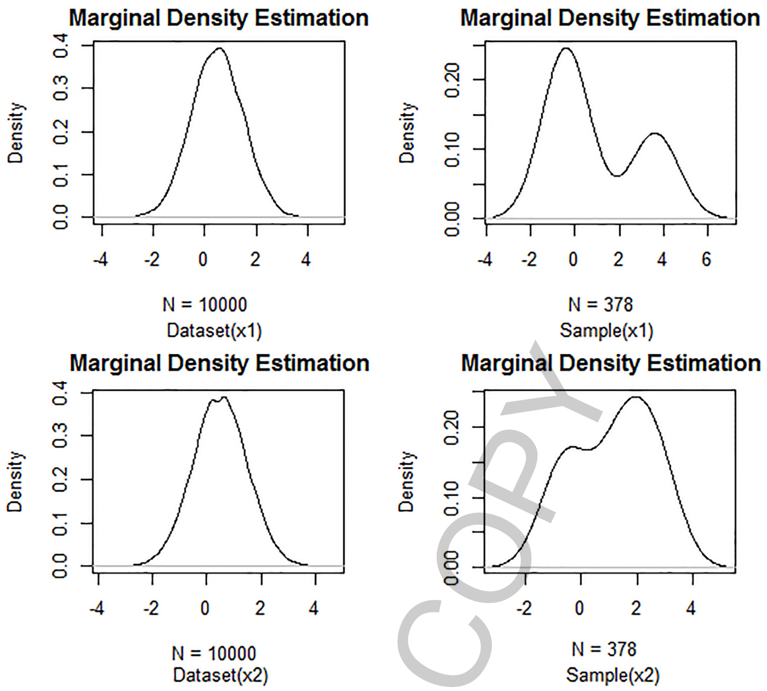


Fig. 12. Marginal PDFs after executing FRPS given a Gaussian dataset in $\Re^2$ with $N = 10000$.

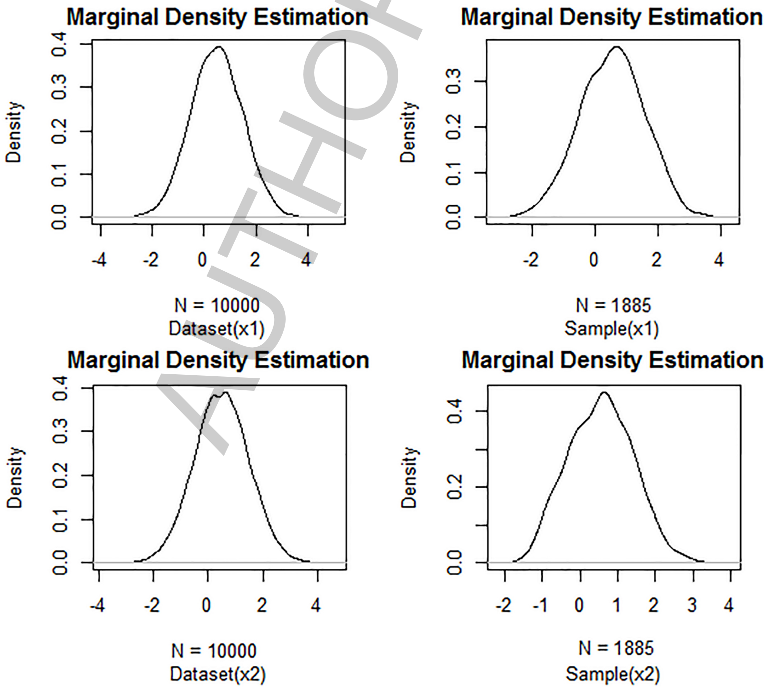


Fig. 13. Marginal PDFs after executing FRIS given a Gaussian dataset in $\Re^2$ with $N = 10000$.

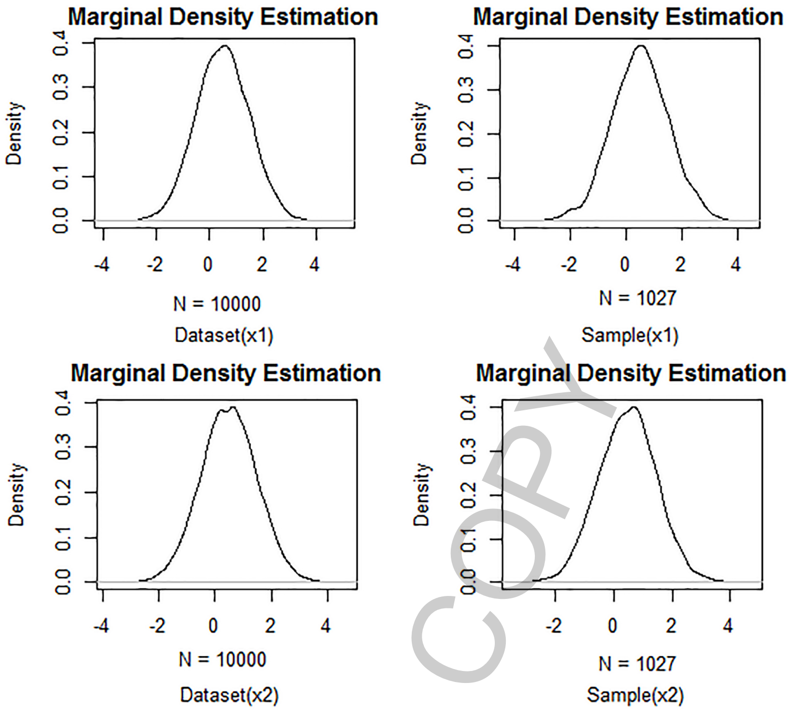


Fig. 14. Marginal PDFs after executing FRIS given a Gaussian dataset in $\Re^2$ with $N = 10000$.

obtained with FRPS and FRIS respectively). Based on the above, we can hypothesize that our method allows us to obtain the sample with the optimal size subject to constraints that force the preservation of statistical properties characterized by its PDF.

Since the above results are not enough to ensure the effectiveness of our method (at least from the statistical view point), we executed FRPS and FRIS with the same datasets used in Subsection 4.2, in order to obtain the average values of the effectiveness measures SS, $error_\mu$ and $error_\sigma$. The obtained results are shown in Table 6.

We can see that the baseline methods achieved to reduce about 90% of the original dataset in contrast with our method that achieved a reduction about 80% in average. However, the values of $error_\mu$ and $error_\sigma$ show that our method achieved to retain the statistical properties of the original dataset with an error ratio about 1%. Meanwhile, the baseline methods achieved error ratios about 14% and 24%. It means that a significant reduction of the sample size does not imply that this sample is optimal. Our method allows us to find the sample with the minimal size bounded by its capability to retain the statistical properties of the original data. In this regard, our method outperforms the effectiveness of the baseline methods.

5. Conclusions

A new reduction data method based on the entropy has been defined in order to find a minimal sample that preserves the information of a large dataset. Finding the optimal sample involves an optimization problem that requires an efficient method to explore the huge feasible space. We use EGA as the best alternative. A first approach allows us to verify that our method is able to find a sample from a dataset that retains the original properties characterized by the probability distribution. Based on these results a wide set of experiments on synthetic datasets was executed. We found that in general, the sample retains about 98% of the statistical properties of P and its size is about 30% of the original data size. Since often the datasets have hidden information which are not possible to be characterized through statistical measures. We wanted to ensure that the reduced dataset retains such information. We measured this information through the so-called classification ability of the sample relative to the original dataset. We used a suite of datasets that represent classification problems, in order to show that our method obtains the optimal sample that are able to train BC as well as the original data. Based on the results, we can tackle many applications in DM and ML that require data reduction. As future work, we want to show that our method can be applied to those problems related to feature selection, where removing redundant attributes is compulsory.

Appendix A: Eclectic genetic algorithm

For those familiar with the methodology of genetic algorithms it should come as no surprise that a number of questions relative to the best operation of the algorithm immediately arose. The Simple Genetic Algorithm [50] frequently mentioned in the literature leaves open the optimal values of, at least, the following parameters:

- Probability of crossover (P_c).
- Probability of mutation (P_m).
- Population size.

Additionally, premature and/or slow convergence are also of prime importance. For this EGA incorporates the following:

1. Full elitism over the last set of Θ candidate solutions. Given that, by iteration t, the number of solutions tested is Θt, the set of candidates in such generation consists of the best Θ solutions (individuals).
2. Deterministic selection as opposed to the traditional proportional selection operator. Such scheme emphasizes genetic variety by imposing a strategy that enforces crossover of predefined individuals. After sorting the individual's fitness from better to worse, the i^{th} individual is combined with the $(\Theta - i)^{th}$ individual.
3. Crossover is performed with a probability P_c. Annular crossover makes this operation position independent. Annular crossover allows for unbiased building block search, a central feature to GA's strength. Two randomly selected individuals are represented as two rings (the parent individuals). Semi-rings of equal size are selected and interchanged to yield a set of offspring. Each parent contributes the same amount of information to their descendants.
4. Mutation is performed with probability P_m. Mutation is uniform and, thus, is kept at very low levels. For efficiency purposes, we do not work with mutation probabilities for every independent bit. Rather, we work with the expected number of mutations which, statistically is equivalent to calculating mutation probabilities for every bit. Hence, the expected number of mutations is calculated from $\ell * \Theta * P_m$, where ℓ is the length of the genome in bits and Θ is the number of individuals in the population.

Algorithm 3: Eclectic Genetic Algorithm

```
Data:
$\Theta$ = Number of candidate solutions
$C = \emptyset$, set of candidate solutions
$c_i = 0$, the $i^{th}$ candidate
$\vec{f} = \emptyset$, fitness array of the candidates solutions
$P_c$ = Crossover probability
$P_m$ = Mutation probability
$\ell$ = Length of the Individual
$b2m = \ell * \Theta * P_m$ number of bits to mutate
Result: The top 1 candidate which encodes the best optimal solution of a given problem
Generate a random set $C$ of candidate solutions whose length is $\ell$.
$C$ = initialization($\Theta$, $\ell$);
Determine the fitness for each $c_i \in C$ based on objective function:
$f$ = evaluate($C$);
Sort candidates from best to worst based on their fitness:
sort($C$, $f$);
while convergence criteria are not met do
    duplicate($C$);
    $bottom = 2 * \Theta$;
    for $i = \Theta$ to $2 * \Theta$ do
        Generate a random number $R$;
        if $R > P_c$ then
            crossover($c_i$, $c_{bottom}$));
        end
        $bottom = bottom - 1$
    end
    Mutate the population in $b2m$ randomly selected bits:
    mutate($C$);
    $f$ = evaluate($C$);
    sort($C$, $f$);
    Eliminate the worst $\Theta$ individuals from $C$
    $C$ = remove($C$);
    Return top 1 from $C$
end
```

References

[1] P. Mukhopadhyay, Theory and methods of survey sampling, PHI Learning Pvt. Ltd., 2009.

[2] P.V. Sukhatme, Sampling theory of surveys with applications.

[3] K. Black, Business statistics: for contemporary decision making, John Wiley & Sons, 2011.

[4] W.G. Cochran, Sampling techniques, John Wiley & Sons, 2007.

[5] W. Feller, An introduction to probability theory and its applications, Vol. 2, John Wiley & Sons, 2008.

[6] C.E. Shannon, A mathematical theory of communication, *ACM SIGMOBILE Mobile Computing and Communications Review* **5**(1) (2001), 3–55.

[7] F. Glover, Tabu search-part i, *ORSA Journal on Computing* **1**(3) (1989), 190–206.

[8] S.P. Brooks and B.J. Morgan, Optimization using simulated annealing, *The Statistician* (1995), 241–257.
[9] M. Dorigo and M. Birattari, Ant colony optimization, in: Encyclopedia of machine learning, Springer, 2010, pp. 36–39.
[10] J. Kennedy, Particle swarm optimization, in: Encyclopedia of Machine Learning, Springer, 2010, pp. 760–766.
[11] W.M. Spears, K.A. De Jong, T. Bäck, D.B. Fogel and H. De Garis, An overview of evolutionary computation, in: Machine Learning: ECML-93, Springer, 1993, pp. 442–459.
[12] S.A. Geritz, G. Mesze, J.A. Metz et al., Evolutionarily singular strategies and the adaptive growth and branching of the evolutionary tree, *Evolutionary Ecology* **12**(1) (1998), 35–57.
[13] J.-H. Kim and H. Myung, Evolutionary programming techniques for constrained optimization problems, Evolutionary Computation, *IEEE Transactions on* **1**(2) (1997), 129–140.
[14] J.R. Koza, F.H. Bennett III and O. Stiffelman, Genetic programming as a darwinian invention machine, in: Genetic Programming, Springer, 1999, pp. 93–108.
[15] D.E. Goldberg and J.H. Holland, Genetic algorithms and machine learning, *Machine Learning* **3**(2) (1988), 95–99.
[16] G. Rudolph, Convergence analysis of canonical genetic algorithms, Neural Networks, *IEEE Transactions on* **5**(1) (1994), 96–101.
[17] A.F. Kuri-Morales, E. Aldana-Bobadilla and I. López-Peña, The best genetic algorithm ii, in: Advances in Soft Computing and Its Applications, Springer, 2013, pp. 16–29.
[18] A.K. Morales and C.V. Quezada, A universal eclectic genetic algorithm for constrained optimization, in: Proceedings of the 6th European congress on intelligent techniques and soft computing, Vol. 1, 1998, pp. 518–522.
[19] T.-J. Yao, C.B. Begg and P.O. Livingston, Optimal sample size for a series of pilot trials of new agents, *Biometrics* (1996), 992–1001.
[20] P. Müller, G. Parmigiani, C. Robert and J. Rousseau, Optimal sample size for multiple testing: the case of gene expression microarrays, *Journal of the American Statistical Association* **99**(468) (2004), 990–1001.
[21] S. Walter, M. Eliasziw and A. Donner, Sample size and optimal designs for reliability studies, *Statistics in Medicine* **17**(1) (1998), 101–110.
[22] H. Liu and H. Motoda, Instance selection and construction for data mining, Vol. 608, Springer Science & Business Media, 2013.
[23] K.C. Gowda and G. Krishna, The condensed nearest neighbor rule using the concept of mutual nearest neighborhood, *IEEE Transactions on Information Theory* **25**(4) (1979), 488–490.
[24] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, Systems, Man and Cybernetics, *IEEE Transactions on* (3) (1972), 408–421.
[25] D.G. Lowe, Similarity metric learning for a variable-kernel classifier, *Neural Computation* **7**(1) (1995), 72–85.
[26] D. Kibler and D. Aha, Learning representative exemplars of concepts: An initial case study, in: Proc. of the 4th International Workshop on Machine Learning, 1987, pp. 24–30.
[27] D.R. Wilson and T.R. Martinez, Reduction techniques for instance-based learning algorithms, *Machine Learning* **38**(3) (2000), 257–286.
[28] K. Kalegele, H. Takahashi, J. Sveholm, K. Sasai, G. Kitagata and T. Kinoshita, On-demand data numerosity reduction for learning artifacts, in: Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on, IEEE, 2012, pp. 152–159.
[29] D.B. Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms, in: Proceedings of the eleventh international conference on machine learning, 1994, pp. 293–301.
[30] C.R. Reeves and D.R. Bush, Using genetic algorithms for training data selection in rbf networks, in: Instance selection and construction for data mining, Springer, 2001, pp. 339–356.
[31] J.R. Cano, F. Herrera and M. Lozano, Using evolutionary algorithms as instance selection for data reduction in kdd: An experimental study, Evolutionary Computation, *IEEE Transactions on* **7**(6) (2003), 561–575.
[32] Z. Pawlak, Rough sets, *International Journal of Computer & Information Sciences* **11**(5) (1982), 341–356.
[33] R. Jensen and C. Cornelis, Fuzzy-rough instance selection, in: Fuzzy Systems (FUZZ), 2010 IEEE International Conference on, IEEE, 2010, pp. 1–7.
[34] N. Verbiest, C. Cornelis and F. Herrera, Frps: A fuzzy rough prototype selection method, *Pattern Recognition* **46**(10) (2013), 2770–2782.
[35] H. Oja, Multivariate nonparametric methods with R: an approach based on spatial signs and ranks, Springer Science & Business Media, 2010.
[36] R.J. Hyndman, The problem with sturges rule for constructing histograms, Monash University.
[37] R.J. Hyndman and Y. Fan, Sample quantiles in statistical packages, *The American Statistician* **50**(4) (1996), 361–365.
[38] D.P. Doane, Aesthetic frequency classifications, *The American Statistician* **30**(4) (1976), 181–183.
[39] D.M. Lane, Online statistics education: An interactive multimedia course of study, http://onlinestatbook.com/2/graphing_distributions/histograms.html, accessed: 2015-12-03 (2015).
[40] D.J. White and G. Anandalingam, A penalty function approach for solving bi-level linear programs, *Journal of Global Optimization* **3**(4) (1993), 397–419.

[41] A.F. Kuri-Morales and J. Gutiérrez-García, Penalty function methods for constrained optimization with genetic algorithms: A statistical analysis, in: MICAI 2002: Advances in Artificial Intelligence, Springer, 2002, pp. 108–117.
[42] K. Sayood, Introduction to data compression, Newnes, 2012.
[43] M. Lichman, Uci machine learning repository, abalone dataset, http://archive.ics.uci.edu/ml/datasets/Abalone, accessed: 2015-03-22 (1995).
[44] M. Lichman, Uci machine learning repository, cars dataset, http://archive.ics.uci.edu/ml/datasets/Car+Evaluation, accessed: 2015-03-22 (1997).
[45] M. Lichman, Uci machine learning repository, census income dataset, http://archive.ics.uci.edu/ml/datasets/Census+Income, accessed: 2015-03-22 (1996).
[46] M. Lichman, Uci machine learning repository, hepatitis dataset, http://archive.ics.uci.edu/ml/datasets/Hepatitis, accessed: 2015-03-22 (1988).
[47] M. Lichman, Uci machine learning repository, yeast dataset, http://archive.ics.uci.edu/ml/datasets/Yeast, accessed: 2015-03-22 (1996).
[48] A. Agresti, Categorical Data Analysis, Vol. 359, John Wiley & Sons, 2002.
[49] L.F. Shampine, R.C. Allen and S. Pruess, Fundamentals of numerical computing, John Wiley, 1997.
[50] J. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence, 2a ed., MIT Press, 1992.