Information Sciences 580 (2021) 221–242

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

OCEAn: Ordinal classification with an ensemble approach

Belén Vega-Márquez *, Isabel A. Nepomuceno-Chamorro, Cristina Rubio-Escudero, José C. Riquelme

Department of Computer Languages and Systems, University of Sevilla, Sevilla, Spain

ARTICLE INFO

Article history:
Received 22 November 2020
Received in revised form 18 August 2021
Accepted 22 August 2021
Available online 24 August 2021

Keywords:
Ordinal classification
Ensemble optimization
Weighting-vote method cost-sensitive
Genetic algorithm

ABSTRACT

Generally, classification problems catalog instances according to their target variable without considering the relation among the different labels. However, there are real problems in which the different values of the class are related to each other. Because of interest in this type of problem, several solutions have been proposed, such as cost-sensitive classifiers. Ensembles have proven to be very effective for classification tasks; however, as far as we know, there are no proposals that use a genetic-based methodology as the metaheuristic to create the models. In this paper, we present OCEAn, an ordinal classification algorithm based on an ensemble approach, which makes a final prediction according to a weighted vote system. This weighted voting takes into account weights obtained by a genetic algorithm that tries to minimize the cost of classification. To test the performance of this approach, we compared our proposal with ordinal classification algorithms in the literature and demonstrated that, indeed, our approach improves on previous results.

1. Introduction

The vast majority of classification algorithms predict a categorical variable and do not consider a possible order relation among the labels. This is one of the great challenges that exist today, because a great variety of problems exist in which the entries to be classified maintain a positional or ordinal dependency that conditions the best solution. The discipline that addresses this type of problem is called "ordinal classification" or "ordinal regression" (the terms are used interchangeably). This technique attempts to solve this problem by considering a set of ordered labeled data, in which each instance is a pair key-value $(x, y) \in X \times Y$ that follows an unknown distribution $P(x, y)$, where x is an input vector that belongs to a space X, and y is a label within a finite space of ordered labels $Y = \{C_1, C_2, \ldots, C_k\}$ [25]. These labels can take any value, even numeric, but there is no meaningful numeric difference among them [17]. The goal of ordinal classification is to find a classifier or a function $h : X \rightarrow Y$ that predicts the value of y using the knowledge obtained from x. The effectiveness of these algorithms is measured through a loss function $L(y, \hat{y})$ that determines the cost of predicting $\hat{y}$ when the actual value is y. This cost function takes into account that there is a natural order between the labels to correctly guide the algorithm being used to solve the problem. This order can be expressed by $C_1 \prec C_2 \prec \ldots \prec C_k$, where $\prec$ expresses a relation of order. Ordinal classification has been successfully applied in several works, as in [35], where an order-preserving tree-generation algorithm was used to solve multi-attribute classification problems with k linearly ordered classes.

* Corresponding author.
E-mail address: bvega@us.es (B. Vega-Márquez).

https://doi.org/10.1016/j.ins.2021.08.081
0020-0255/

Today, ensembles are considered one of the most precise methods of solving problems in the area of machine learning, as can be seen in [37]. The main idea of ensemble algorithms is to combine multiple classifiers in some way (typically by weighted or unweighted voting) to produce a single classification that is generally found to be more accurate than any of the classifications given by each of the base classifiers alone. The classifiers used to compose the ensemble can be algorithms of any type, such as neural networks, decision trees, and so on. These algorithms take a set of labeled data and produce a model that is later used to label new examples with unknown labels. The main objective of combining the outputs of these classifiers is to compensate for the errors that a classifier can make with the outputs of other base classifiers [37].

The main line of research in the field of ensembles is to find the optimal way to combine the classifiers that compose it. The work carried out in [38] serves as an example on how to combine base classifiers; they propose a new ensemble method based on artificial neural networks, called CCE, to solve multiclass classification problems, where the final output is calculated by averaging the outputs associated with each class and choosing the class with the maximum value.

In this study, we present OCEAn (Ordinal Classification algorithm based on an Ensemble Approach), which makes a final prediction according to a weighting-vote system, with the weights being optimized by means of a genetic algorithm. To verify the performance of this ensemble, we compared it with the experimentation carried out in [19], where datasets from different sources and with different characteristics were used. As listed above, many papers have been published on ordinal classification and ensembles. However, as far as we know, no one has yet combined ensembles to solve classification problems where an intrinsic order exists between class labels by means of gene-based algorithms. Research on the combination of these techniques is promising because genetic algorithms and ensembles have shown very good results in classification tasks [28,36].

The rest of this paper is organized as follows: In Section 2, we introduce related works. In Section 3, we define in detail the OCEAn approach. In Section 4, we explain the datasets used in the study, the methodology used for the comparison, and the experimental results obtained. Finally, in Section 5, we provide our conclusions.

2. Related works

All the techniques used to date for ordinal classification can be organized in 3 broad groups [19] as shown in Fig. 1: naïve approaches, ordinal binary decompositions, and threshold models.

Naïve approaches frame all the algorithms that try to solve ordinal classification problems from a decomposition point of view; that is, the methodology to decomposes the problem in other simpler paradigms making some conjectures. Within this large group, there are 3 approaches: regression, nominal classification, and cost-sensitive classification. First, in regression approaches, the target variable is converted to an integer in order to adjust the ordinal solution by means of a regression technique and subsequently rounded in a postprocessing step. Usually, a relation exists between the actual value assigned and the position of the label within the order of the subset. However, this is not always the best approach, because the returned real values do not fully take into account the integer "order" aspect that underlies an ordinal classification problem. In 2018, a regression model to discover the relation between treatment concerns and the fertility of 1218 woman was proposed in [18]. Second, when the order between the classes is not taken into account and the algorithm that is trained treats the labels as if they were purely nominal labels, this is called nominal classification. The problem with this type of technique is that, by not taking into account the nature of the data, the ordering information is lost, so a larger dataset is needed to extract valuable knowledge. An example of this approach can be seen in [31], where a classification and regression tree (CART) was used to determine the quality of the services in the public transport on the Granada metropolitan transit system in 2007. Third are the cost-sensitive methods. In some problems, in addition to considering the order among labels, it is also important to know that the cost of predicting a y_1 label when the real one is y, is different from the cost of predicting y_2 when the real one is still y. An example of this might be predicting whether a person will commit bank fraud or not. It is worse for the bank when the algorithm predicts that no fraud will be committed and it is actually committed than when the reverse occurs. In cost-sensitive classification algorithms, the costs are reflected in a cost matrix $C(i,j)$, in which the columns of the

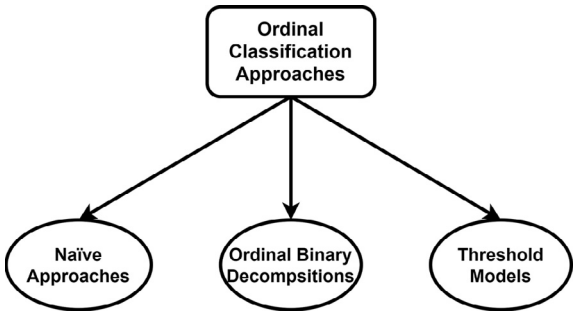

Fig. 1. Ordinal classification approaches.

matrix indicate the actual class and the rows indicate the predicted class or vice versa; each of the positions of the matrix indicates the cost caused by predicting the label corresponding to row i when the actual class is that of column j.

Another large group into which ordinal classification algorithms fall is called **ordinal binary decompositions**. This group is based on the idea of "divide and conquer," because its main feature is that it divides the ordinal label into several binary labels. These models can be subdivided in 2 groups: *multiple model* and *multiple-output single model* approaches. In the first case, a different model is trained for each of the sub-problems. In [13], an ordinal classification problem is solved by independently processing each binary classification problem, and then binary outputs are combined into a predicted label. Still within this group, another approach takes into account a certain order between the target variables. The method consists of converting an ordinal classification problem into separate binary problems, so that it is transformed from a k-class problem to k-1 binary class problems. The first step is to generate as many datasets as k-1 binary problems exist and, once these datasets are generated, apply a classification algorithm to each of the generated data sets. When predicting the label for a new unknown instance, each of the probabilities of the original k ordinal classes is estimated using the k-1 learning models generated, and the one with the highest probability is then selected. In [6], an ordinal hyper-plane ranking algorithm, OHRank, based on multiple-model approach, was proposed. It was applied to the problem of estimating human ages via facial images. In the multiple-output single model, only one model is trained for all sub-problems encountered when dividing up the class. One of the most common techniques used when dealing with multiple-output models is neural networks, as they easily allow multiple responses to be dealt with in a friendly way. In [29], a deep convolutional neural network was used to predict the age of people from photos of their faces; in particular, an ordinal regression problem was transformed into a series of binary classification sub-problems.

Finally, **threshold models** are the best known and most successful models in modeling problems with ordinal characteristics. These approaches take into account that there is a continuous variable that can explain the behavior of the ordinal variable. The main objective of these algorithms is to find a function $f(x)$ that allows the mapping of the continuous values predicted into the ordinal labels. This vast group can be subdivided into 7 subgroups: *cumulative link models*, *support vector machines*, *discriminant learning*, *perceptron learning*, *augmented binary classification*, *ensembles*, and *Gaussian processes*. A deep convolutional neural network model for ordinal regression using cumulative link models in the output layer is proposed in [41]. A support vector machine (SVM) model can be seen in [9], where SVOREX, an SVM with explicit constraints, is used to carry out a classification approach that deals with the remitting behavior of immigrants according to their individual characteristics. A comparison of 3 linear discriminant learning algorithms used for ordinal classification can be found in [5], where the experiments were carried out on synthetic and real datasets. In 2005, a perceptron learning method for ordinal classification was proposed in [10], where they developed a new algorithm, called PRank, to predict user ratings on new films. Furthermore, a new evaluation methodology for the algorithms used for text line segmentation is developed in [4], using an augmented binary classification approach. An ensemble method for ordinal classification in the transportation sector was developed by [49]. Finally, within the group of Gaussian processes, we find the work carried out by [8], where a gene selection algorithm was developed to discover gene expression patters associated with ordinal clinical phenotypes.

Many researchers make use of ensembles in classification tasks because of their high performance. This type of learning is considered the state-of-the art approach for solving not only supervised problems [1], but also unsupervised problems [11]. Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the base learning algorithms alone [30].

Ensembles also owe their good performance to their relation with other well-tested and proven methodologies, such as granular computing [47] or multitask learning [39]. Granular computing is a paradigm of information processing. From a practical perspective, granular computing is considered a way of structured problem solving [48]. In general, it involves 2 operations: granulation and organization. Granulation is based on the decomposition of a whole into different parts, as is the case with the bagging methodology, since it involves random sampling of the data. In contrast, organization takes care of integrating various parts into a whole [34], again reminiscent of bagging, because this algorithm combines the different outputs of the base classifiers to provide a final prediction. In the work carried out in [26], the authors proposed a nature-inspired approach of ensemble learning that was validated through experimental studies by using real-life datasets. They showed that their proposal overcame limitations of the bagging approach. Regarding multitask learning methodologies, in [44], a new multitask ensemble-based learning framework was proposed, whereby the authors demonstrated a new way to manipulate class labels to construct a strong ensemble of classifiers. They generated different new class labels through the Cartesian product, which were used to build a new component classifier for each of them. In addition, a new ensemble decision tree learning algorithm was proposed in [45]. The novelty of this proposal was that each input attribute was considered an extra task to introduce bias in the classification problem. In terms of ordinal regression, several papers use multitask learning; for example, in [24] an SVM algorithm was applied to several ordinal regression scenarios using multitask learning via conic programming.

Ensembles can be classified into 2 categories depending on how the individual outputs of the base algorithm are aggregated [37]: weighting methods and meta-learning methods. In ordinal classification, the idea is usually to adapt existing ensemble algorithms to take into account the ordinal nature of the data being addressed. In [43], an SVM-based ensemble was implemented for ordinal regression, and its effectiveness was demonstrated by applying it to benchmark synthetic datasets. Another interesting paper applying ensembles for ordinal regression can be found in [20], where a modification of the

well-known k-nearest neighbour (knn) method was applied to a problem with an intrinsic order among its instances. They used a weighted majority vote mechanism for the aggregation process.

All of the literature on this subject indicates that better results can be obtained when using the ensemble methodology compared with individual base algorithms. Therefore, we propose a new methodology within this paradigm. To our knowledge, no other paper has used genetic algorithms to determine the way in which the base algorithms are organized to provide an output, which is why we propose this new way of combining the individual classifiers.

3. Proposal

Ensemble methods are learning algorithms that are based on a set of classifiers that label new instances by making a consensus decision from the predictions made by them. One of the possibilities to make that decision is to carry out weighted voting with the predictions proposed by the different classifiers. In this work, we present OCEAn, a new algorithm framed within threshold algorithms, which consists of an ensemble that uses a genetic algorithm to determine the optimal weight for each of the base classifiers that make up the ensemble.

The steps followed in our methodology can be seen in Fig. 2. There are 3 main phases, which will be enumerated below:

1. **Cost-Sensitive Base Classifiers Training:** In the first step, the dataset is divided in 3 parts, the first for training the cost-sensitive base classifiers, called DS_{train}, the second for training the genetic algorithm and obtaining the best weights, DS_{opt}, and the third for validating the proposal, DS_{val}. The training subset is used to train the t base classifiers chosen to compose the ensemble using the matrix cost M. Then, once the classifiers have been trained, 2 matrices of predictions, named C_{opt} and C_{val}, are obtained, with as many rows (as instances) as the optimization and validation subsets contain, respectively, and as many columns as there are models that have been trained. These matrices are the result of predicting with t models, the label for each of the instances of the 2 subsets of data DS_{opt} and DS_{val}, respectively.
2. **Ensemble Optimization:** In the second phase, using C_{opt}, an ensemble optimization approach is used to establish the weights that should be given to each of the models to provide an optimal solution to the classification problem. The optimal solution is that which leads to the minimal cost, using for that purpose the cost matrix M in the error function. The optimization process; that is, the search for the optimal weights, is performed using a genetic algorithm.
3. **Evaluation:** In the third phase, predictions obtained in the first phase on the validation set, C_{val}, are used to apply the weights that conform the ensemble and then evaluate the performance of the proposal.

Each of these steps will be detailed below, as well as some definitions needed to understand the process. First, we explain the necessary steps to obtain the prediction matrices that will be used as input to the genetic algorithm. We will specify the cost-sensitive classifiers and the parameters used. Second, we explain the steps followed by the ensemble optimization algorithm. Third, we will explain the 2 measures used to evaluate the results: mean absolute error (MAE) and mean zero squared error (MZE). Finally, we will show the pseudocode for the algorithm.

3.1. Cost-sensitive base classifiers training

In the first step, the input data to the genetic algorithm are generated. With the aim that our approach takes into account the cost idea, we decided to use cost-sensitive classifiers as the base for the ensemble. A cost-sensitive algorithm uses a cost

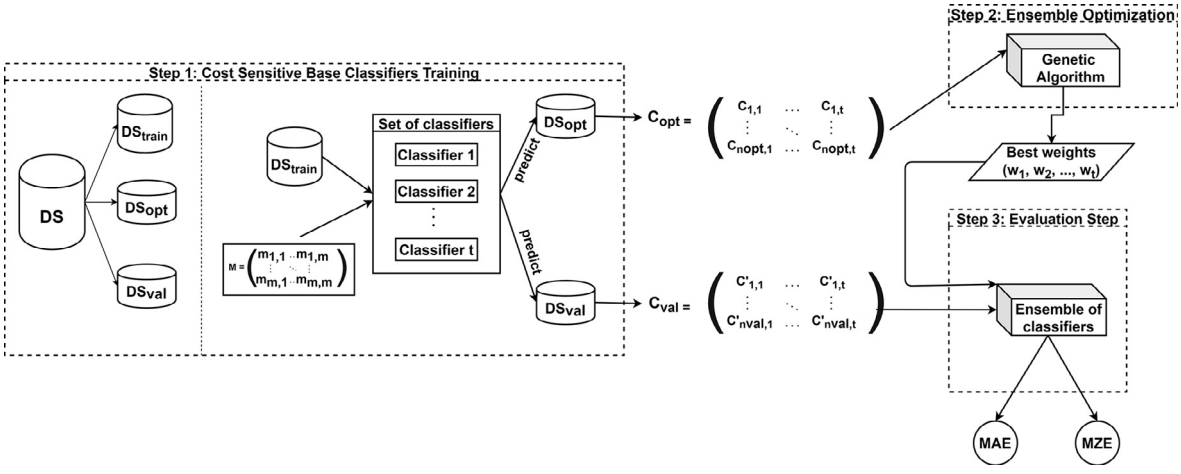

Fig. 2. Steps in the OCEAn proposal.

matrix M to generate a model with the lowest cost. The cost matrix encodes the penalty of classifying samples from one class as another.

To formalize our OCEAn approach, several definitions are required.

Definition 1 (*Input Dataset*). Let $DS = \{(e_1, l_1), \ldots, (e_n, l_n)\}$ be an input instances dataset, where each e_i is an n-tuple of attribute values belonging to a certain instance space X, and l_i is a label from a label set $L = \{1, m\}$ where m is the number of different labels in the dataset DS.

Definition 2 (*Cost Matrix*). Let M be a cost matrix defined as a squared matrix. The element $M(i,j)$ of the matrix represents the cost of predicting label j when the real is i.

Definition 3 (*Cost-Sensitive Base Classifiers*). Let $P = \{p_1, p_2, \ldots, p_i, \ldots, p_t\}$ be a set of t cost-sensitive base classifiers. Each p_i classifier, given an input dataset, estimates the label l_j of each instance e_j, taking into account the cost matrix M.

The first step implies the building of the input data for our proposal. To this aim, the dataset DS is divided in 3 parts, named DS_{train}, DS_{opt}, and DS_{val}. DS_{train} corresponds to the data that will be used in the cost-sensitive classifiers training step. The 2 remaining parts will be used for steps 2 and 3 of our proposal. DS_{opt} is the part of the dataset used for the ensemble optimization process; that is, the data used to find the best weights. Then, DS_{val}, which will be reserved until the end of the process, will be used in the evaluation phase. Let n_{train}, n_{opt}, and n_{val} be the number of instances in DS_{train}, DS_{opt}, and DS_{val}, respectively. The classifiers in P are trained with cost matrix M and the DS_{train} subset. From this training, 2 prediction matrices are obtained: C_{opt} ($n_{opt} \times t$) and C_{val} ($n_{val} \times t$). A C_{ij} term represents the prediction carried out by the classifier j in the instance i (e_i), as shown in Fig. 3:

3.2. Weighting-vote ensemble optimization

It is well known that the objective of ensemble methods is to overcome the results given by individual classifiers. There are several ways to combine the outputs of the cost-sensitive classifiers used to conduct the ensemble. The technique that we used is the weighting-vote method, as described in [14]. With this technique, a specific weight is assigned to each of the classifiers used to build the ensemble, and the weight is expected to be higher for a particular base classifier that performs better.

The main problem with this type of method is assigning the optimal weights to each classifier. This problem can be viewed as an optimization problem [50]. Therefore, we decided to use a genetic algorithm to solve it. The purpose of the proposed genetic-based algorithm, given a subset of data (in this case DS_{opt}), is to find those weights for each of the base classifiers that predict new instances in the best possible way (in this case, those predictions that imply the lowest cost). Five main operations in genetic algorithms are repeated until the stopping criterion G (generation number) is reached. They are described below:

Initialization and coding: In this work, each chromosome represents the weight associated with each base classifier. We have used a positional encoding, where the gene i_{th} represents the weight associated with the i_{th} base classifier. Therefore, chromosome W has the following form:

$$W = (w_1, w_2, \ldots, w_i, \ldots, w_t)$$

where t is the number of base classifiers from P. Each population in the genetic algorithm is composed of N chromosomes.

Evaluation: The evaluation of individuals is carried out on the basis of an error function, F, which calculates, for each of the instances of the dataset, what the cost would be of predicting their label, taking into account the vote carried out with the weights assigned by the individual. Thus, the error function would be the sum of these costs for each of the instances of the dataset. These costs can be calculated using the cost matrix M, the real label, and the predicted label. The goal is to minimize this error function.

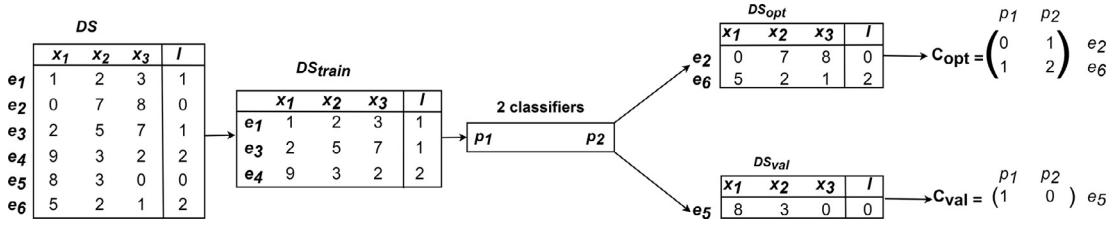

Fig. 3. Example of optimization and validation prediction matrix generation. The dataset is divided in 3 ($DS_{train}, DS_{opt}, DS_{val}$), and then t classifiers are trained with DS_{train}. (Although t can take any number, we are using $t = 2$ for readability reasons). Once trained, predictions are made for each of the instances in DS_{opt} and DS_{val} and each of the classifiers, obtaining the prediction matrices C_{opt} and C_{val}, respectively.

In a formal way, the error function is calculated as follows:

1. Given a chromosome W, a prediction matrix C and an instance e_i for each of the possible labels in the dataset DS, the weight associated with each is calculated as follows:

$$f(W, C, e_i, l) = \sum_{j=1}^{t} w_j [C_{ij} == l], where [C_{ij} == l] = \begin{cases} 1, & \text{if } C_{ij} = l \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Then, given W and C, the winning label L_i is calculated for each instance e_i as follows:

$$L_i = argmax_l f(W, C, e_i, l) \quad (2)$$

We illustrate an example in Fig. 4. Consider $W = (3, 1)$; that is, the p_1 classifier has a weight of 3 and the classifier p_2 has a weight of 1. According to the C_{opt} matrix, p_1 predicted label 0 for the instance e_2 and p_2 predicted label 1 for the same instance. As a consequence, the value for $f(W, C_{opt}, e_2, 0)$ is 3, because classifier p_1 (with weight 3) predicted label 0. The value for $f(W, C_{opt}, e_2, 1)$ is 1, because the classifier that predicted label 1 has a weight of 1 in W. No classifier predicted label 2 and therefore $f(W, C_{opt}, e_2, 2)$ is 0. Thus, the winning label L_2 is 0.

2. Finally, the error function associated with a specific individual W and a prediction matrix C belonging to a dataset DS are the calculated as the sum of the costs, according the cost matrix M, for each example in DS:

$$F(W, C, DS) = \sum_{i=1}^{n} M(l_i, L_i) \quad (3)$$

where l_i is the actual label for i_{th} instance. An example can be seen in Fig. 5. In the figure, step 1 is applied to all rows in the prediction matrix C_{opt}, giving the winning label for each of the instances. Then, taking into account the cost matrix M and the real label of the 3 instances, the error function is calculated. The error function sums up, for each of the instances, the cost of predicting the resulting label in step 1 (L_i) when the real one is l_i.

Selection: The methodology used to make the selection is the deterministic tournament method. This technique runs several tournaments among a few chromosomes chosen randomly from the population. The number of chromosomes that participate in a tournament is called selection pressure, represented by sel_{press}, which determines the probability that individuals with worse error take part in the tournament. In each tournament, the winning chromosome is the one with a lowest error value.

Reproduction and crossover. Uniform crossover was used, because it is one of the most effective methods when looking for all possibilities that can arise when the parents are recombined. In this type of crossover, each gene in the offspring has the same probability of belonging to one or the other parent. The number of children generated by 2 fathers is called n_h.

Mutation. Nonuniform mutation was used in this paper. For each chromosome in the population, this mutation operator takes into account a mu_{press} pressure, so that each individual in the new population has a probability of a gene mutation of mu_{press}%. The new gene is randomly generated from a chosen interval.

3.3. Evaluation step

Once the best weights are obtained after the optimization process, these weights are applied to the prediction matrix over the validation set. This allows us to obtain a prediction, which is used to calculate the validation measures, in this case, MAE and MZE.

MAE is the average of the absolute differences between prediction and actual observation, and MZE is the error rate of the classifier:

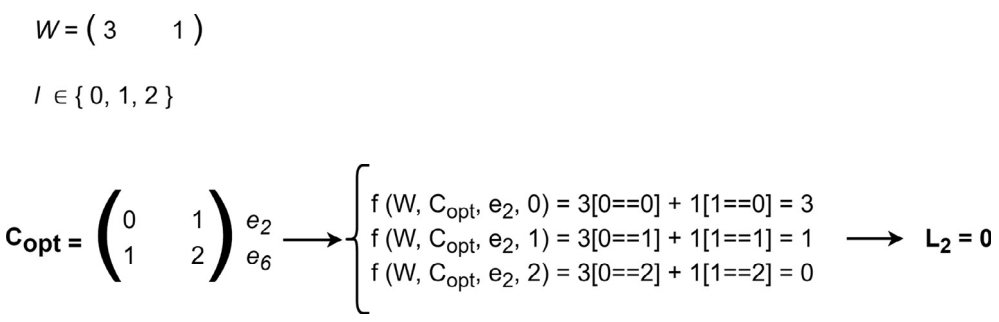

Fig. 4. Step 1 example. The winning label for the first row in C_{opt} is calculated by taking into account the set of weights W. It can be seen that the winning label is 0, because it is the one with the maximum weight.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \tag{4}$$

$$MZE = \frac{1}{n}\sum_{j=1}^{n}|y_j \neq \hat{y}_j| = 1 - Accuracy \tag{5}$$

where y_i is the true label and $\hat{y}_i$ is the predicted label. It should be noted that, in ordinal classification, when the labels are not numeric, MAE and MZE are calculated by first converting the labels to numerical values based on the cost matrix.

To make the evaluation step more reliable, our proposal took into account several splits of the data, which led to a number of iterations of the algorithm; that is, the OCEAn proposal was trained and validated with each of the subsets resulting from each partition, and the resulting measure is the average of the results obtained with each iteration.

3.4. Pseudocode

The pseudocode for our proposal is given in Algorithm 1.

Algorithm 1. OCEAn Algorithm

Input : dataset DS, cost matrix M, set of classifiers P, classifiers number t, population size N, number of iterations IT, generations number G, selection pressure sel_{press}, number of children in each reproduction $n_{children}$, mutation pressure mu_{press}

Output: Mean Absolute Error MAE_{total}, Mean-Zero Squared Error MZE_{total}

$it, n, g, MAE_{total}, MZE_{total} \leftarrow 0$
$IP \leftarrow \emptyset$ //(Initial Population)
while $it < IT$ **do**
  $MAE, MZE \leftarrow 0$
  $FP(FinalPopulation), H(Children) \leftarrow \emptyset$
  //Step 1: Cost-Sensitive Base Classifiers Training
  $DS_{train}, l_{train}, DS_{opt}, l_{opt}, DS_{val}, l_{val} =$ train_test_split(DS)
  trained_classifiers = train_algorithms(DS_{train}, P, M)
  C_{opt} = make_predictions($DS_{opt}, trained\_classifiers$)
  C_{val} = make_predictions($DS_{val}, trained\_classifiers$)
  //Step 2: Ensemble Optimization
  while $n < N$ **do**
    $W \leftarrow \emptyset$
    W = initialize_random_chromosome(t)
    $IP \leftarrow IP \cup \{W\}$
    $n = n + 1$
  end
  $FP \leftarrow IP$
  while $g < G$ **do**
    $list\_parents$ = tournament_selection($FP, sel_{press}, C_{opt}, M, l_{opt}$)
    $FP' \leftarrow list\_parents$
    while $len(FP') < N$ **do**
      $parents$ = random_sample($list\_parents$)
      H = uniform_crossover($parents, n_{children}$)
      $FP' = FP' \cup \{H\}$
    end
    FP' = mutate_population(FP', mu_{press})
    $FP \leftarrow FP'$
  end
  //Step 3: Evaluation
  $bestWeights = \arg\max_{W \in FP} F(W, C_{opt}, DS_{opt})$
  y_{pred} = get_predictions($bestWeights, C_{val}$)
  MAE_{it}, MZE_{it} = evaluate(y_{pred}, l_{val})
  $MAE = MAE + MAE_{it}$
  $MZE = MZE + MZE_{it}$
end
$MAE_{total} = MAE/IT$
$MZE_{total} = MZE/IT$

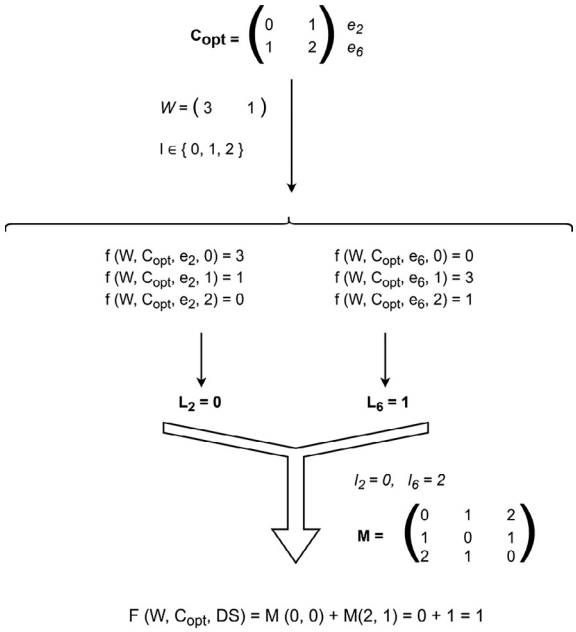

Fig. 5. In this figure, the error function is calculated for C_{opt}. Step 1 is repeated as many times as the number of rows in the prediction matrix, in this case 2 times. Once the winning label has been obtained for each row, the cost of predicting the winning label is calculated considering the cost matrix *M*.

4. Results

In this section, we describe the experiments that were carried out to analyze the performance of the proposal. First, selected datasets are described. Second, algorithm parameters are explained. Third, the software and hardware infrastructure used is specified. Fourth, the design of the comparison carried out is explained. Finally, a statistical analysis to test the effectiveness of our proposal is included.

4.1. Data description

To show the power of the ensemble classifier, predictions were conducted based on the same datasets used in a previous work from Gutiérrez et al. [19] to which we compared our proposal. The datasets can be divided in 2 groups: real ordinal classification datasets extracted from 2 benchmark repositories (UCI [12] and the mldata.org repository [33]) and regression datasets retrieved from [7], which are converted into ordinal classification datasets using equal frequency discretization; that is, discretizing the target variable into different bins with the same number of instances in each bin. All of these data can be found at http://www.uco.es/grupos/ayrna/orreview, provided by the team that conducted the research in [19]. It is necessary to point out that these sets of data do not belong to ordinal classification problems, they have a regression function, and they do not present the same characteristics as those related to ordinal classification, such as data unbalance. This allowed us to check how our approach behaved in both balanced and unbalanced datasets. In addition, we used a different number of bins for the same dataset, so we could see whether the size of the intervals influenced the final results.

We used a total of 41 datasets, 17 of which were real ordinal classification datasets. The remaining 24 were those discretized from the 12 datasets found in [7]. Two different discretizations were made, with 5 and 10 target classes for each dataset. Tables 1 and 2 show the characteristics of each of the datasets; that is, the name, number of instances, number of attributes (total number of attributes, taking into account the class attribute as well), and number of classes. The first contains the real ordinal classification datasets, and the second the discretized datasets.

4.2. Algorithm settings

One of the fundamental aspects in this research was the choice of the base algorithms to be used for the ensemble and of the parameters for the genetic algorithm. The choices made will be described below.

Table 1
Real ordinal classification datasets. Although many of these datasets have not been used in the literature as ordinal classification problems, the fact that they have a categorical label can be used to establish an ordering relation among them. These datasets were extracted from UCI [12] and the mldata.org repository [33]

Dataset name	# Instances	# Attributes	# Classes
contact-lenses (CL)	24	5	3
pasture (PA)	36	23	3
squash-stored (SS)	52	25	3
squash-ustored (SU)	52	24	3
tae (TA)	151	6	3
newthyroid (NT)	215	6	3
balance-scale (BS)	625	5	3
SWD (SW)	1000	11	4
car (CA)	1728	7	4
bondrate (BO)	57	11	5
toy (TO)	300	3	5
eucalyptus (EU)	736	20	5
LEV (LE)	1000	5	5
automobile (AU)	205	26	6
winequality-red (WR)	1599	12	6
ESL (ES)	488	5	9
ERA (ER)	1000	5	9

Table 2
Discretized ordinal classification datasets. This group of datasets has the peculiarity that they are not datasets intended for ordinal classification, but are regression datasets. However, by means of a discretization process, the label can be transformed into Q different bins with equal frequency and they become suitable for ordinal classification problems. Specifically, for each of the original datasets, 2 binarizations were applied, one with 5 labels and the other with 10. The original data can be retrieved from [7].

Dataset name	# Instances	# Attributes	# Classes
pyrim5 (P5)	74	27	5
machine5 (M5)	209	7	5
housing5 (H5)	506	14	5
stock5 (S5)	950	10	5
abalone5 (A5)	4177	11	5
bank5 (B5)	8192	9	5
bank5' (BB5)	8192	33	5
computer5 (C5)	8192	13	5
computer5' (CC5)	8192	22	5
cal.housing5 (CH5)	20640	9	5
census5 (CE5)	22784	9	5
census5' (CEE5)	22784	17	5
pyrim10 (P10)	74	27	10
machine10 (M10)	209	7	10
housing10 (H10)	506	14	10
stock10 (S10)	950	10	10
abalone10 (A10)	4177	11	10
bank10 (B10)	8192	9	10
bank10' (BB10)	8192	33	10
computer10 (C10)	8192	13	10
computer10' (CC10)	8192	22	10
cal.housing10 (CH10)	20640	9	10
census10 (CE10)	22784	9	10
census10' (CEE10)	22784	17	10

4.2.1. Cost-sensitive base algorithms

The main tool used to carry out this proposal was Weka, given its wide use in the literature [42] and its ease of use. Weka has a cost-sensitive classification algorithm, called *"CostSensitiveClassifier"*, which allows any classification algorithm (called base classifier) to be cost-sensitive. This can be done in 2 different ways: by re-weighting training instances according to the total cost assigned to each class or by predicting the target value with minimum expected misclassification cost [46]. In this case, the second option was used, because the intention was to maintain the same importance for each of the instances without the class interfering in it. This method also needs a cost matrix to be able to carry out ordinal classification, and the experiment took into account numerous datasets each with a different number of classes. Therefore, we opted for the simplest solution in terms of cost matrix; that is, we considered the one in which the position m_{ij} of the matrix M is equal to the

Table 3
Example of different cost matrices for 3- and 4-class classification problems.

3-class cost matrix	4-class cost matrix
$\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{pmatrix}$

difference between the categories represented by position (i, j); that is, $m_{ij} = |i - j|$. Table 3 shows 2 examples of cost matrices for 3- and 4-class classification problems.

The base algorithms used for training *CostSensitiveClassifier* are enumerated in Table 4. These algorithms were chosen from Weka with the premise that they could handle multiclass labels, which is why we chose the 18 algorithms presented in the table below. For each category, we selected the best known methods. All were trained with the default parameters provided by the tool.

4.2.2. Weighting-vote ensemble optimization

As previously noted, a genetic algorithm was the methodology used to establish the best weights in the weighting vote ensemble. There are 5 basic phases in genetic algorithms, as first stated in [21]: initialization, evaluation, selection, reproduction and crossover, and mutation. Because one of the fundamental aspects within genetic algorithms is the parameters used, a parameter search was conducted. First, we fixed the population size to 200 ($N = 200$), then we made a grid search (Table 5) for the 2 datasets with the fewest instances for both truly ordinal and discrete datasets; that is, contact lenses (CL) and pyrim5 (P5), and then we kept the parameters that provided better results. The parameters chosen for each of the phases are listed below and can be seen in Table 6.

1. **Initialization:** The initial population consisted of 200 individuals ($N = 200$), each of which was made up of 18 elements (one for each of the trained base algorithms) of random integers, as done in [40] between 1 and 10, in order to cover a wider spectrum of possible combinations among the 18 classifiers. This avoids the situation in which the dominant classifiers always obtained the best weights, with the goal that all algorithms take part in the optimization process.
2. **Evaluation:** The evaluation of individuals was carried out on the basis of an error function that calculates, for each instance of the dataset, what the cost would be of predicting a label, taking into account the vote carried out with the weights assigned by the individual in question, so that the error function would be the sum of these costs for each instance of the dataset. The goal is to minimize this error function.
3. **Selection:** The methodology used to make the selection was the deterministic tournament method. To be able to explore new regions in the search space, a pressure of 2 was considered; that is, in each tournament, 2 individuals participate ($sel_{press} = 2$) and the one with the lowest error is considered to be in the next generation.
4. **Reproduction and crossover:** Uniform crossover was used. The number of children generated from 2 parents is 2; that is, $n_{children} = 2$.

Table 4
Base learners used in *CostsensitiveClassification* Weka algorithm.

Algorithm Family	Algorithm Name
Functions	Logistic (Log) MultilayerPerceptron (MLP) SMO
Bayes	NaivesBayes (NB) BayesNet (BN)
Lazy	KStar IBk
Meta	AdaBoostM1 (AdB) Bagging (Bag) LogitBoost (LB) ClassificationViaRegression (CVR)
Trees	J48 DecisionStump (DS) LMT RandomForest (RF) REPTree
Rules	PART JRip

Table 5
Grid search for the training parameters of OCEAn.

Parameter	Grid Search
sel_{press}	[2, 4, 6]
mu_{press}	[0.2, 0.4, 0.6]
G	[40, 60, 80, 100]

Table 6
Parameters used to conduct the genetic algorithm.

Parameter	Number
N	200
sel_{press}	2
mu_{press}	0.4
$n_{children}$	2
G	40

5. **Mutation:** Mutation takes into account a mu_{press} pressure of 0.4, so that each individual in the new population has a probability of a gene mutation of 40 %. The new gene will be randomly generated from the interval between 1 and 100.

The stop criterion is the number of generations, G; in this case, we chose $G = 40$; that is, for each of the iterations, 40 repetitions of the genetic algorithm were conducted. We also took into account 10 different seeds to make up the ensemble, to reduce the influence of the initial population used to carry out the optimization process. For that purpose, the number of iterations was used as the initial seed in the generation of the initial population. All of the parameters chosen are listed in Table 6.

4.3. Software and experimental setting

The cost-sensitive base classifiers used to carry out the ensemble were implemented with the Python Weka Wrapper, which allowed us to use Weka from within Python. It also provides a wrapper around the basic functionality of Weka. Weka is software for machine learning written in Java, developed at the University of Waikato. This is free software licensed under the GNU General Public License. The optimization process and the evaluation step were carried out with the Python programming language. Because of the large number of experiments, this work made intensive use of the High Performance Computing (HPC) cluster at Centro Informático Científico de Andalucía (CICA) formed by more than 60 nodes, with a total of almost 650 cores, with 32 calculation nodes (372 cores) connected to each other through Infiniband DDR at 20 Gbps. The source code with the different tests performed in this study can be found in https://github.com/bvegaus/OCEAn.

4.4. Experimental results

In this subsection, we explain how the experimental setting was designed and the results obtained. In the survey carried out in [19], 16 algorithms were used to assess the performance obtained by each one of them. To make the comparison with our proposal, we decided to keep the algorithms that stood out in terms of error in the survey carried out in [19]; that is, SVC1V1 [22], SVMOP [13,43], SVORIM [9], and SVOREX [9], as well as a simple voting algorithm to serve as a baseline. In addition, for a more extensive and fair comparison, we added the results obtained with a model based on granular computation presented in [26].

To make the comparison and validation more reliable, we used the same methodology as in [19], where random splits were carried out: one part to train the algorithms and the remainder to test the classifiers. Because these splits only considered 2 partitions (D_{train}, D_{opt}) and we needed 3 ($DS_{train}, DS_{opt}, DS_{val}$), we kept their D_{opt} as our DS_{val} subset in both types of datasets, ordinal and discretized. For our DS_{train} and DS_{opt} subsets we needed to split their D_{train} subset into 2 parts. Depending on whether the dataset was ordinal or discretized, we used different approaches. For ordinal regression datasets, 66% was considered for DS_{train} and the remaining 34% for DS_{opt} (in this way we had the same number of instances for optimization and validation). For the discretized regression datasets, 50% was considered for DS_{train} and the remaining 50% for the DS_{opt} subset. This decision was made because in some datasets, the number of instances for validation was higher than that for training, so if we kept the same proportion used in the ordinal regression datasets, only a few instances would be used for the training of classifiers and the genetic algorithm (steps 1 and 2 of the proposal). We carried out 5 random splits.

The comparison between our proposal and [19] can be seen in Tables 7 and 8 for real ordinal datasets, and in Tables 9 and 10 for ordinal discretized datasets. For each dataset, we show the results achieved with SVC1V1, SVMOP, SVORIM, SVOREX, granular computing Model (GC), simple voting (SimpleVot) and results obtained with the OCEAn algorithm, for both MAE

Table 7
MAE comparison for real ordinal datasets.

	MAE						
	SVC1V1	SVMOP	SVORIM	SVOREX	GC	SimpleVot	OCEAn
CL	$0.500_{0.000}$	$0.500_{0.000}$	$0.500_{0.000}$	$0.500_{0.000}$	$\mathbf{0.300_{0.247}}$	$0.300_{0.218}$	$0.303_{0.179}$
PA	$0.555_{0.078}$	$0.911_{0.318}$	$0.666_{0.000}$	$0.577_{0.092}$	$0.733_{0.278}$	$\mathbf{0.267_{0.1}}$	$0.268_{0.105}$
SS	$0.738_{0.042}$	$0.707_{0.323}$	$0.569_{0.042}$	$0.738_{0.042}$	$0.553_{0.100}$	$0.385_{0.095}$	$\mathbf{0.373_{0.064}}$
SU	$0.369_{0.114}$	$0.369_{0.034}$	$0.538_{0.000}$	$0.384_{0.143}$	$0.476_{0.147}$	$0.262_{0.069}$	$\mathbf{0.249_{0.046}}$
TA	$0.684_{0.110}$	$1.005_{0.143}$	$0.668_{0.014}$	$0.626_{0.065}$	$\mathbf{0.600_{0.100}}$	$0.632_{0.082}$	$0.637_{0.038}$
NT	$0.296_{0.000}$	$0.385_{0.130}$	$0.296_{0.000}$	$0.296_{0.000}$	$0.077_{0.065}$	$0.033_{0.009}$	$\mathbf{0.030_{0.007}}$
BS	$0.156_{0.036}$	$1.173_{0.920}$	$0.115_{0.028}$	$0.156_{0.036}$	$0.266_{0.049}$	$0.143_{0.033}$	$\mathbf{0.109_{0.029}}$
SW	$0.534_{0.021}$	$0.804_{0.154}$	$0.580_{0.044}$	$0.543_{0.049}$	$0.460_{0.033}$	$0.430_{0.025}$	$\mathbf{0.425_{0.019}}$
CA	$0.389_{0.014}$	$0.198_{0.081}$	$0.231_{0.012}$	$0.388_{0.007}$	$0.081_{0.003}$	$0.083_{0.009}$	$\mathbf{0.038_{0.009}}$
BO	$0.600_{0.094}$	$0.693_{0.111}$	$0.600_{0.094}$	$0.600_{0.094}$	$1.080_{0.098}$	$0.587_{0.179}$	$\mathbf{0.558_{0.151}}$
TO	$1.141_{0.011}$	$1.162_{0.170}$	$0.952_{0.011}$	$1.141_{0.011}$	$0.394_{0.064}$	$0.192_{0.066}$	$\mathbf{0.154_{0.049}}$
EU	$1.136_{0.137}$	$1.768_{0.471}$	$0.572_{0.062}$	$1.233_{0.104}$	$0.842_{0.125}$	$0.362_{0.033}$	$\mathbf{0.359_{0.026}}$
LE	$0.656_{0.038}$	$0.863_{0.101}$	$0.545_{0.023}$	$0.647_{0.019}$	$0.434_{0.023}$	$0.431_{0.015}$	$\mathbf{0.412_{0.022}}$
AU	$1.115_{0.023}$	$0.850_{0.097}$	$1.015_{0.016}$	$1.115_{0.023}$	$1.584_{0.229}$	$0.346_{0.087}$	$\mathbf{0.330_{0.061}}$
WI	$0.462_{0.011}$	$0.477_{0.023}$	$0.470_{0.030}$	$0.491_{0.048}$	$0.869_{0.056}$	$0.409_{0.01}$	$\mathbf{0.379_{0.020}}$
ES	$0.519_{0.017}$	$\mathbf{0.309_{0.031}}$	$0.375_{0.008}$	$0.490_{0.014}$	$0.547_{0.076}$	$0.341_{0.013}$	$0.340_{0.009}$
ER	$1.776_{0.004}$	$1.536_{0.180}$	$1.261_{0.050}$	$1.776_{0.004}$	$1.327_{0.061}$	$1.222_{0.048}$	$\mathbf{1.220_{0.053}}$

The best result obtained for each of the datasets is highlighted in bold type.

Table 8
MZE comparison for real ordinal datasets.

	MZE						
	SVC1V1	SVMOP	SVORIM	SVOREX	GC	SimpleVot	OCEAn
CL	$0.333_{0.000}$	$0.333_{0.000}$	$0.333_{0.000}$	$0.333_{0.000}$	$0.200_{0.139}$	$0.200_{0.075}$	$\mathbf{0.193_{0.099}}$
PA	$0.533_{0.049}$	$0.622_{0.230}$	$0.666_{0.000}$	$0.577_{0.092}$	$0.533_{0.197}$	$\mathbf{0.267_{0.100}}$	$0.268_{0.105}$
SS	$0.584_{0.042}$	$0.553_{0.191}$	$0.569_{0.042}$	$0.584_{0.042}$	$0.446_{0.126}$	$0.338_{0.088}$	$\mathbf{0.327_{0.060}}$
SU	$0.353_{0.116}$	$0.338_{0.042}$	$0.538_{0.000}$	$0.384_{0.143}$	$0.400_{0.147}$	$0.262_{0.069}$	$\mathbf{0.249_{0.046}}$
TA	$0.589_{0.044}$	$0.757_{0.075}$	$0.668_{0.014}$	$0.568_{0.047}$	$\mathbf{0.447_{0.067}}$	$0.511_{0.040}$	$0.507_{0.033}$
NT	$0.296_{0.000}$	$0.270_{0.057}$	$0.296_{0.000}$	$0.296_{0.000}$	$0.059_{0.035}$	$0.033_{0.009}$	$\mathbf{0.030_{0.007}}$
BS	$0.119_{0.018}$	$0.628_{0.460}$	$0.104_{0.022}$	$0.119_{0.018}$	$0.174_{0.024}$	$0.116_{0.018}$	$\mathbf{0.095_{0.020}}$
SW	$0.518_{0.025}$	$0.662_{0.059}$	$0.557_{0.036}$	$0.523_{0.044}$	$0.434_{0.029}$	$0.416_{0.025}$	$\mathbf{0.416_{0.019}}$
CA	$0.287_{0.009}$	$0.184_{0.071}$	$0.206_{0.011}$	$0.292_{0.004}$	$0.059_{0.002}$	$0.071_{0.005}$	$\mathbf{0.034_{0.006}}$
BO	$\mathbf{0.413_{0.029}}$	$0.480_{0.073}$	$0.413_{0.029}$	$0.413_{0.029}$	$0.693_{0.076}$	$0.453_{0.12}$	$0.437_{0.094}$
TO	$0.706_{0.000}$	$0.682_{0.082}$	$0.736_{0.005}$	$0.706_{0.000}$	$0.210_{0.041}$	$0.181_{0.058}$	$\mathbf{0.154_{0.049}}$
EU	$0.641_{0.044}$	$0.825_{0.125}$	$0.490_{0.034}$	$0.672_{0.033}$	$0.470_{0.048}$	$0.341_{0.024}$	$\mathbf{0.337_{0.023}}$
LE	$0.571_{0.017}$	$0.631_{0.078}$	$0.496_{0.015}$	$0.571_{0.010}$	$0.401_{0.021}$	$0.402_{0.018}$	$\mathbf{0.381_{0.022}}$
AU	$0.680_{0.010}$	$0.596_{0.067}$	$0.734_{0.008}$	$0.680_{0.010}$	$0.565_{0.072}$	$0.262_{0.067}$	$\mathbf{0.253_{0.042}}$
WI	$0.419_{0.005}$	$0.429_{0.020}$	$0.430_{0.024}$	$0.443_{0.033}$	$0.488_{0.022}$	$0.374_{0.007}$	$\mathbf{0.347_{0.020}}$
ES	$0.434_{0.021}$	$\mathbf{0.303_{0.028}}$	$0.331_{0.014}$	$0.406_{0.017}$	$0.345_{0.006}$	$0.318_{0.018}$	$0.319_{0.014}$
ER	$0.819_{0.001}$	$0.796_{0.025}$	$0.754_{0.024}$	$0.819_{0.001}$	$\mathbf{0.732_{0.031}}$	$0.747_{0.024}$	$0.747_{0.027}$

The best result obtained for each of the datasets is highlighted in bold type.

and MZE, with the standard deviation in the 5 executions as sub-indexes. The best results obtained among the 7 algorithms are highlighted in bold.

Tables 7 and 8 show the results obtained for real ordinal classification datasets, the first shows the results in terms of MAE, and the second in terms of MZE. In the case of MAE, our proposal obtained better results in 13 out of 17 experiments, and considering MZE, our proposal was better in 12 cases.

Table 9 illustrates the MAE results obtained for the discretized ordinal classification datasets (considering 5 and 10 bins). In this case, our proposal performed better in 17 of the 24 datasets. The MZE results in Table 10 showed better results in 15 datasets with our approach.

A graphical summary of these results can be seen in Figs. 6 and 7, where we can observe the better performance of our proposal compared with previous results.

To obtain more information about the performance of our proposal, the confusion matrices of the CA dataset were added as an example (Table 11). In the table, 5 confusion matrices are shown, one for each of the iterations of each dataset (recall that each of the training steps for each dataset was composed of 5 iterations, each with a different division among training, optimization, and validation data).

As can be seen, most of the values are on the diagonal of the matrices, which would be optimal, and there are no values at the bottom left and top right, which means that, for this dataset, our algorithm never predicted a label with a maximum cost. This phenomenon can be seen in the F1-Score measure, which has a value of 0.909 for the CA dataset.

Table 9
MAE comparison for discretized datasets.

	MAE						
	SVC1V1	SVMOP	SVORIM	SVOREX	GC	SimpleVot	OCEAn
P5	$1.075_{0.313}$	$1.658_{0.459}$	$1.166_{0.000}$	$1.116_{0.194}$	$1.316_{0.234}$	$1.042_{0.213}$	$\mathbf{0.964_{0.116}}$
M5	$0.545_{0.078}$	$1.176_{0.174}$	$0.650_{0.054}$	$0.647_{0.075}$	$0.728_{0.192}$	$\mathbf{0.444_{0.070}}$	$0.457_{0.066}$
H5	$0.625_{0.053}$	$1.627_{0.291}$	$0.626_{0.045}$	$1.018_{0.068}$	$0.907_{0.119}$	$\mathbf{0.399_{0.076}}$	$0.411_{0.058}$
S5	$0.362_{0.045}$	$1.472_{0.127}$	$0.337_{0.007}$	$0.351_{0.020}$	$0.386_{0.555}$	$0.133_{0.017}$	$\mathbf{0.122_{0.012}}$
A5	$0.852_{0.055}$	$1.393_{0.093}$	$0.782_{0.019}$	$1.061_{0.026}$	$1.101_{0.072}$	$0.702_{0.008}$	$\mathbf{0.685_{0.006}}$
B5	$0.972_{0.075}$	$1.355_{0.238}$	$1.200_{0.000}$	$0.805_{0.034}$	$1.589_{0.049}$	$0.694_{0.065}$	$\mathbf{0.624_{0.082}}$
BB5	$1.593_{0.150}$	$1.570_{0.085}$	$1.200_{0.000}$	$1.400_{0.000}$	$1.912_{0.034}$	$1.147_{0.034}$	$\mathbf{1.138_{0.023}}$
C5	$0.588_{0.058}$	$1.549_{0.417}$	$0.731_{0.066}$	$0.748_{0.100}$	$1.368_{0.091}$	$\mathbf{0.573_{0.028}}$	$0.716_{0.022}$
CC5	$0.723_{0.027}$	$1.385_{0.116}$	$0.645_{0.006}$	$0.779_{0.042}$	$1.530_{0.129}$	$\mathbf{0.492_{0.036}}$	$0.765_{0.040}$
CH5	$1.007_{0.083}$	$1.449_{0.182}$	$1.173_{0.030}$	$1.002_{0.112}$	$1.197_{0.106}$	$0.748_{0.022}$	$\mathbf{0.719_{0.036}}$
CE5	$1.040_{0.033}$	$1.645_{0.082}$	$1.014_{0.125}$	$1.013_{0.092}$	$1.245_{0.057}$	$0.724_{0.047}$	$\mathbf{0.716_{0.020}}$
CEE5	$1.058_{0.070}$	$1.552_{0.177}$	$1.178_{0.020}$	$1.070_{0.116}$	$1.503_{0.039}$	$0.781_{0.033}$	$\mathbf{0.766_{0.041}}$
P10	$\mathbf{2.050_{0.240}}$	$3.358_{0.323}$	$2.566_{0.086}$	$2.608_{0.133}$	$3.033_{0.555}$	$2.367_{0.336}$	$2.185_{0.346}$
M10	$1.440_{0.196}$	$3.216_{0.542}$	$1.318_{0.126}$	$1.332_{0.115}$	$2.464_{0.250}$	$1.014_{0.113}$	$\mathbf{0.989_{0.092}}$
H10	$1.395_{0.135}$	$2.866_{0.144}$	$1.106_{0.059}$	$2.293_{0.273}$	$3.018_{0.140}$	$0.949_{0.076}$	$\mathbf{0.929_{0.087}}$
S10	$0.813_{0.019}$	$2.841_{0.348}$	$0.704_{0.022}$	$1.788_{0.075}$	$1.416_{0.142}$	$0.321_{0.017}$	$\mathbf{0.286_{0.014}}$
A10	$1.861_{0.141}$	$2.804_{0.248}$	$1.529_{0.032}$	$2.601_{0.087}$	$2.919_{0.064}$	$1.481_{0.028}$	$\mathbf{1.431_{0.021}}$
B10	$2.272_{0.225}$	$2.877_{0.194}$	$2.324_{0.107}$	$1.782_{0.252}$	$4.217_{0.101}$	$1.626_{0.155}$	$\mathbf{1.600_{0.122}}$
BB10	$2.840_{0.114}$	$3.426_{0.248}$	$2.500_{0.000}$	$2.742_{0.097}$	$4.437_{0.043}$	$2.376_{0.097}$	$\mathbf{2.358_{0.074}}$
C10	$1.289_{0.065}$	$2.622_{0.346}$	$1.337_{0.055}$	$1.490_{0.177}$	$3.848_{0.120}$	$\mathbf{1.208_{0.060}}$	$1.509_{0.038}$
CC10	$2.304_{0.181}$	$2.797_{0.542}$	$1.271_{0}.045$	$2.274_{0.092}$	$3.723_{0.159}$	$\mathbf{1.052_{0.058}}$	$1.612_{0.035}$
CH10	$2.248_{0.096}$	$2.929_{0.213}$	$1.789_{0.078}$	$2.064_{0.223}$	$3.570_{0.239}$	$1.677_{0.080}$	$\mathbf{1.568_{0.049}}$
CE10	$2.083_{0.173}$	$2.804_{0.258}$	$1.760_{0.102}$	$2.335_{0.314}$	$3.726_{0.120}$	$1.542_{0.046}$	$\mathbf{1.506_{0.041}}$
CEE10	$2.164_{0.069}$	$3.001_{0.198}$	$1.820_{0.091}$	$2.257_{0.203}$	$4.203_{0.102}$	$1.667_{0.069}$	$\mathbf{1.615_{0.039}}$

The best result obtained for each of the datasets is highlighted in bold type.

Table 10
MZE comparison for discretized datasets.

	MZE						
	SVC1V1	SVMOP	SVORIM	SVOREX	GC	SimpleVot	OCEAn
P5	$0.633_{0.068}$	$0.825_{0.061}$	$0.791_{0.000}$	$0.666_{0.051}$	$0.708_{0.083}$	$0.666_{0.078}$	$\mathbf{0.623_{0.053}}$
M5	$0.474_{0.033}$	$0.749_{0.030}$	$0.606_{0.052}$	$0.467_{0.048}$	$0.508_{0.047}$	$\mathbf{0.407_{0.053}}$	$0.409_{0.055}$
H5	$0.460_{0.022}$	$0.819_{0.030}$	$0.561_{0.037}$	$0.598_{0.026}$	$0.492_{0.050}$	$\mathbf{0.343_{0.051}}$	$0.355_{0.039}$
S5	$0.307_{0.043}$	$0.800_{0.027}$	$0.336_{0.006}$	$0.346_{0.018}$	$0.235_{0.029}$	$0.133_{0.017}$	$\mathbf{0.122_{0.012}}$
A5	$0.573_{0.009}$	$0.805_{0.015}$	$0.607_{0.009}$	$0.609_{0.005}$	$0.616_{0.018}$	$0.540_{0.009}$	$\mathbf{0.530_{0.008}}$
B5	$0.652_{0.036}$	$0.768_{0.029}$	$0.800_{0.000}$	$0.627_{0.046}$	$0.706_{0.010}$	$0.56_{0.042}$	$\mathbf{0.522_{0.054}}$
BB5	$\mathbf{0.751_{0.024}}$	$0.780_{0.015}$	$0.800_{0.000}$	$0.800_{0.000}$	$0.792_{0.003}$	$0.759_{0.013}$	$0.755_{0.011}$
C5	$\mathbf{0.455_{0.013}}$	$0.800_{0.069}$	$0.642_{0.010}$	$0.570_{0.043}$	$0.647_{0.017}$	$0.488_{0.024}$	$0.546_{0.014}$
CC5	$\mathbf{0.532_{0.016}}$	$0.730_{0.048}$	$0.610_{0.005}$	$0.555_{0.010}$	$0.684_{0.029}$	$0.438_{0.035}$	$0.579_{0.011}$
CH5	$0.644_{0.023}$	$0.794_{0.028}$	$0.796_{0.005}$	$0.630_{0.041}$	$0.653_{0.024}$	$0.572_{0.007}$	$\mathbf{0.551_{0.017}}$
CE5	$0.616_{0.005}$	$0.809_{0.018}$	$0.744_{0.045}$	$0.631_{0.017}$	$0.642_{0.008}$	$0.553_{0.020}$	$\mathbf{0.547_{0.014}}$
CEE5	$0.636_{0.011}$	$0.811_{0.022}$	$0.794_{0.006}$	$0.663_{0.013}$	$0.702_{0.011}$	$0.592_{0.009}$	$\mathbf{0.580_{0.012}}$
P10	$0.825_{0.054}$	$0.966_{0.034}$	$0.900_{0.037}$	$0.850_{0.063}$	$\mathbf{0.808_{0.104}}$	$0.858_{0.092}$	$0.838_{0.088}$
M10	$0.752_{0.019}$	$0.928_{0.032}$	$0.769_{0.045}$	$0.718_{0.061}$	$0.738_{0.039}$	$0.641_{0.023}$	$\mathbf{0.635_{0.009}}$
H10	$0.694_{0.025}$	$0.876_{0.011}$	$0.700_{0.013}$	$0.811_{0.009}$	$0.757_{0.025}$	$0.575_{0.026}$	$\mathbf{0.575_{0.026}}$
S10	$0.512_{0.017}$	$0.910_{0.020}$	$0.619_{0.014}$	$0.745_{0.013}$	$0.457_{0.007}$	$0.297_{0.020}$	$\mathbf{0.268_{0.017}}$
A10	$0.750_{0.009}$	$0.899_{0.006}$	$0.775_{0.015}$	$0.780_{0.009}$	$0.810_{0.003}$	$0.743_{0.006}$	$\mathbf{0.736_{0.005}}$
B10	$0.859_{0.015}$	$0.902_{0.011}$	$0.890_{0.007}$	$0.805_{0.021}$	$0.882_{0.007}$	$0.783_{0.023}$	$\mathbf{0.779_{0.023}}$
BB10	$\mathbf{0.861_{0.006}}$	$0.9023_{0.002}$	$0.900_{0.000}$	$0.879_{0.010}$	$0.898_{0.001}$	$0.875_{0.014}$	$0.876_{0.013}$
C10	$\mathbf{0.679_{0.007}}$	$0.876_{0.029}$	$0.787_{0.011}$	$0.731_{0.010}$	$0.847_{0.013}$	$0.687_{0.013}$	$0.759_{0.013}$
CC10	$0.838_{0.013}$	$0.878_{0.047}$	$\mathbf{0.767_{0.016}}$	$0.858_{0.010}$	$0.836_{0.008}$	$0.641_{0.016}$	$0.787_{0.004}$
CH10	$0.848_{0.006}$	$0.906_{0.004}$	$0.848_{0.011}$	$0.834_{0.012}$	$0.850_{0.016}$	$0.782_{0.008}$	$\mathbf{0.759_{0.007}}$
CE10	$0.799_{0.009}$	$0.890_{0.012}$	$0.833_{0.015}$	$0.813_{0.012}$	$0.862_{0.004}$	$0.765_{0.017}$	$\mathbf{0.759_{0.013}}$
CEE10	$0.817_{0.010}$	$0.899_{0.008}$	$0.834_{0.011}$	$0.834_{0.008}$	$0.884_{0.006}$	$0.793_{0.007}$	$\mathbf{0.788_{0.004}}$

The best result obtained for each of the datasets is highlighted in bold type.

4.5. Comparison with base classifiers

In addition to comparing our proposal with methods in the literature, we decided to see how well the base algorithms performed separately and confirm whether our ensemble exhibited improvement in the results. One of the advantages of using ensemble-based algorithms is that they can eliminate the difficult decision of which algorithm to choose when faced with a given problem. An ensemble algorithm brings together all the features of the base classifiers it uses to give a final prediction. As proof that our proposal indeed had an advantage over the base algorithms that composed it, we calculated

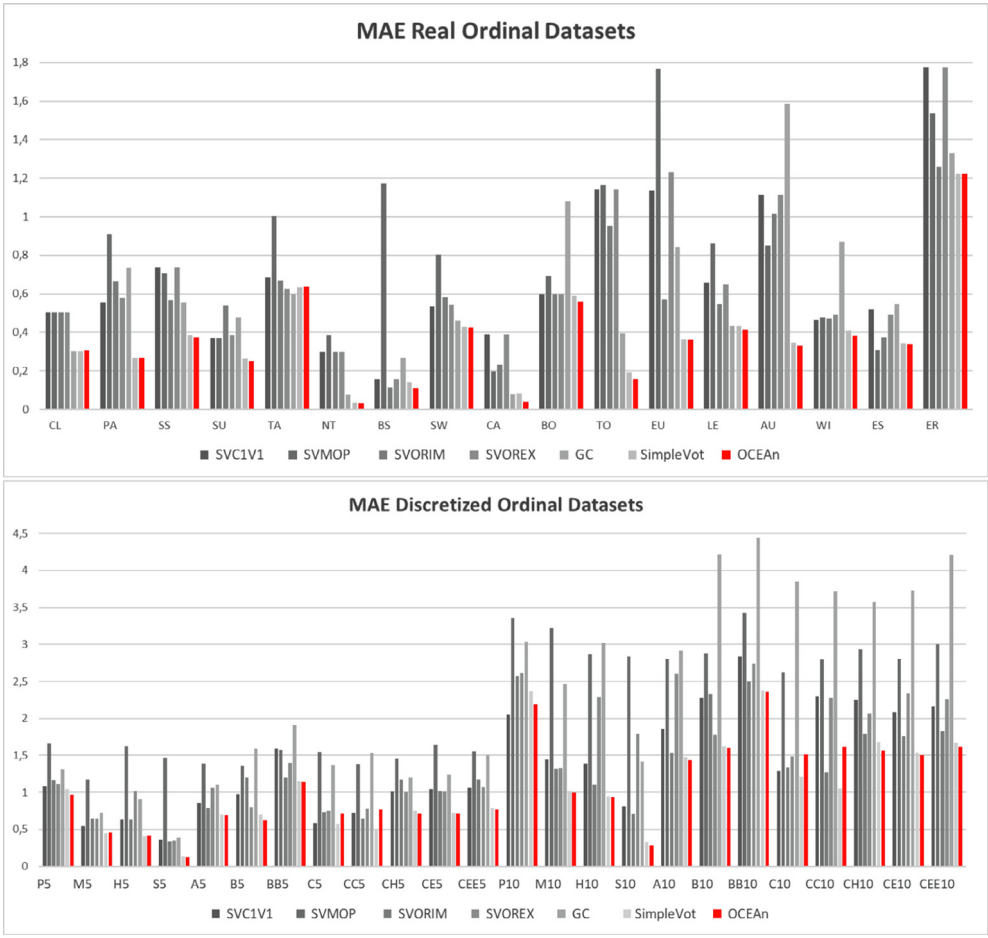

Fig. 6. Comparison of MAE between previous results and our approach. All datasets, real and generated are included.

the metrics individually for the 18 algorithms that made up our ensemble, and then calculated, for each dataset, the average of the 18 classifiers and compared it to the results obtained by OCEAn, as shown in Figs. 8 and 9.

The specific results obtained for each individual classifier and dataset can be found in the Appendix section.

4.6. Statistical test

Because the development or modification of algorithms is a relatively easy task, there is a need for rigorous statistical analysis of published results [16]. Therefore, to test the performance of the OCEAn approach, we applied a statistical framework. The recommended approach for carrying out the statistical comparison of multiple algorithms over multiple datasets is the nonparametric Friedman test [3], along with the Holm post hoc procedure for detecting pairwise differences between 2 algorithms within a multiple comparison test [15]. This statistical analysis was carried out using the open-source platform StatService [32].

The Friedman test could be used to determine whether the MZE and MAE scores were significantly different from the mean rank expected under the null hypothesis [27]. If the Friedman test rejected the null hypothesis with an $\alpha = 0.05$, it means the measures's average ranks were significantly different, so a post hoc test must be carried out to evaluate the relative performance of the proposed algorithms against a control algorithm (the one with the best value; in this case, the one with the minimum value).

Table 12 shows the Friedman test rankings for every algorithm in the comparison, distinguishing between MAE and MZE. The ranking shows, in both cases, that OCEAn ranks first, with scores of 1.5122 and 1.6463 respectively. The Friedman statistic was 70.829 in terms of MAE and 49.002 if we consider MZE scores, with 4 degrees of freedom in both cases.

As previously stated, the level of significance considered for the Friedman test was $\alpha = 0.05$, which means that if the p-value for Friedman was lower than this, significant differences exist among the algorithms compared, because the null hypothesis, which states that they all behave in a similar way, is rejected. In both cases (MAE and MZE rankings), the p-value was lower than 0.00001 indicating that significant differences existed among the approaches.

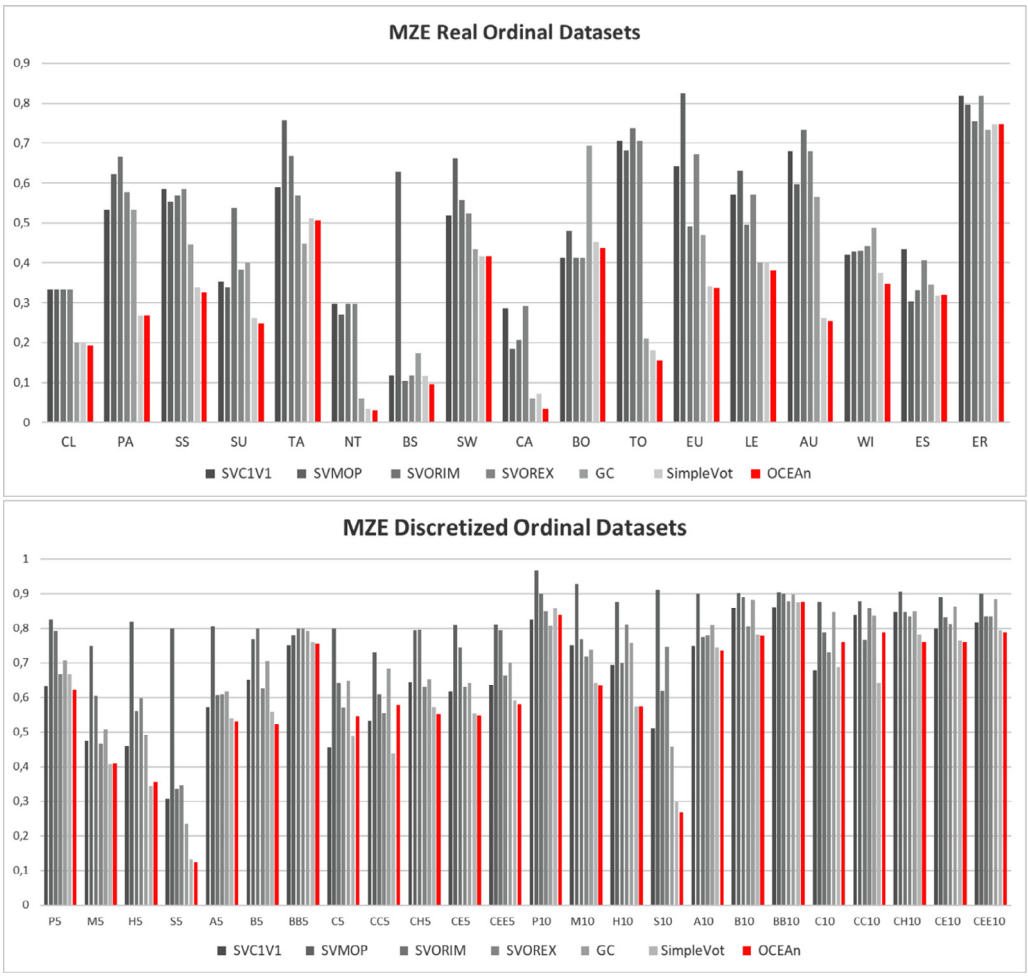

Fig. 7. Comparison of MZE between previous results and our approach. All datasets, real and generated are included.

Table 11
Confusion matrices for the CA (Car) dataset. Each of the tables represents 1 of the 5 divisions of the datasets. Our methodology divided each of the datasets into 3 partitions to carry out both classification with the base algorithms and optimization of the genetic algorithm. These partitions and the entire methodology were done 5 times so that there was no relationship between the data partitions and the final results

Confusion Matrices CA Dataset

	unacc	acc	good	vgood
unacc	301	2	0	0
acc	1	94	1	0
good	0	2	15	0
vgood	0	2	3	11

	unacc	acc	good	vgood
unacc	300	3	0	0
acc	4	89	3	0
good	0	4	13	0
vgood	0	0	1	15

	unacc	acc	good	vgood
unacc	302	1	0	0
acc	9	87	0	0
good	0	3	14	0
vgood	0	0	0	16

	unacc	acc	good	vgood
unacc	293	8	2	0
acc	1	94	0	1
good	0	3	13	1
vgood	0	3	0	13

	unacc	acc	good	vgood
unacc	301	2	0	0
acc	6	89	1	0
good	0	2	15	0
vgood	0	1	3	12

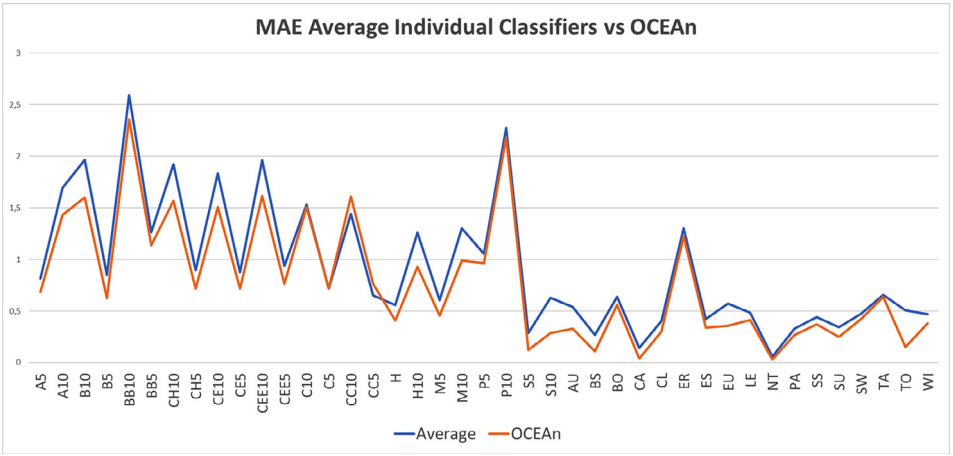

Fig. 8. Comparison of MAE between our proposal and the average of the 18 results obtained with each base classifier for each dataset. It can be seen that our proposal (orange line) performed better in 39 of 41 datasets.

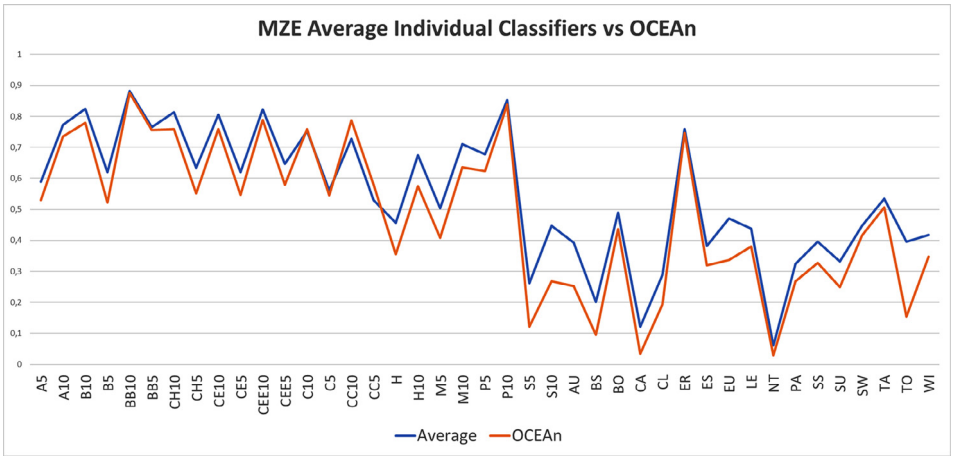

Fig. 9. Comparison of MZE between our proposal and the average of the 18 results obtained with each base classifier for each dataset. It can be seen that our proposal (orange line) performed better in 38 of 41 datasets.

Table 12
Sorted mean ranking obtained by Friedman test.

Ranking			
MAE		MZE	
OCEAn	1.5122	OCEAn	1.6463
SimpleVot	1.9390	SimpleVot	2.1829
SVORIM	3.9146	SVC1V1	3.8415
SVC1V1	4.3537	GC	4.3415
SVOREX	4.6585	SVOREX	4.7317
GC	5.5488	SVORIM	5.1951
SVMOP	6.0732	SVMOP	6.0610

Table 13 presents the results for Conover's post hoc test with Holm-Bonferroni correction. It shows the Conover statistic (T-stat), α_{Holm}, and differences among algorithms, indicating which is the best-ranked method, OCEAn in this case. In this post hoc method, the null hypothesis is rejected if the p-value of the competitors is lower than the α_{Holm}. It can be seen that all the values were lower except for that of the SimpleVot algorithm. This means that even if our algorithm is ranked first, we cannot say with certainty that it is significantly different from the simple voting algorithm. Even so, looking at the results obtained for each of the datasets, we can conclude that our proposal is an approximation that improves, in many cases, on the simple voting algorithm.

Table 13
Post hoc analysis using Conover's post hoc test and Holm-Bonferroni correction.

MAE				MZE			
Algorithm	T-Stat	α_{Holm}	Different ($p < 0.05$) from	Algorithm	T-Stat	α_{Holm}	Different ($p < 0.05$) from
SVC1V1	5.946	<0.001	OCEAn, GC, SimpleVot, SVMOP	SVC1V1	4.602	<0.001	OCEAn, SimpleVot, SVMOP, SVOREX, SVORIM
SVMOP	9.545	<0.001	OCEAn, SimpleVot, SVC1V, SVOREX, SVORIM	SVMOP	9.255	<0.001	OCEAn, SimpleVot, GC, SVC1V1, SVOREX, SVORIM
SVORIM	5.027	<0,001	OCEAn, GC, SimpleVot, SVMOP, SVOREX	SVORIM	7.440	<0.001	OCEAn, SimpleVot, GC, SVC1V1, SVMOP
SVOREX	6.584	<0.001	OCEAn, GC, SimpleVot, SVMOP, SVORIM	SVOREX	6.468	<0.001	OCEAn, SimpleVot, SVC1V1, SVMOP
GC	8.447	<0.001	OCEAn, SimpleVot, SVC1V, SVOREX, SVORIM	GC	5.650	<0.001	OCEAn, SimpleVot, SVMOP, SVORIM
SimpleVot	0.893	1	GC, SVC1V1, SVMOP, SVOREX, SVORIM	SimpleVot	1.125	1	GC, SVC1V1, SVMOP, SVOREX, SVORIM

5. Conclusions

One of the most studied techniques within the field of data science is supervised learning and all the algorithms encompassed by this term. The function of classification algorithms is to predict, given an instance of data, the class of that instance. However, most of these algorithms do not take into account an ordinal relationship between these labels, although there are increasing numbers of problems in everyday life in which this aspect is relevant.

Due to the excellent performance of ensemble algorithms in classification techniques, we decided to use this technique in the current paper. An ensemble classifier is formed by a set of base classifiers whose individual outcomes are combined in some way, typically through a weighted voting, to provide a final response in classifying a query sample. In this study, we present OCEAn, an ensemble of 18 cost-sensitive classifiers to address the problem of ordinal classification. This ensemble performs weighted voting of the predictions obtained by the classifiers that compose it. However, the task of assigning weights is not trivial; the weights were obtained using a genetic algorithm in an attempt to minimize the cost of classification.

To analyze the performance of our proposal, we compared it with the work done in [19], in which they surveyed a vast number of techniques for ordinal classification and carried out an extensive experimentation. This experimentation was based on the training of previous classifiers in the literature on 41 datasets that could be divided in 2 types, depending on whether they were ordinal or not: 17 of the 41 datasets had an ordinal target, and the remaining 24 had a numerical class that was transformed into an ordinal through discretization. The 2 measures used to carry out the comparison were MAE and MZE.

The results obtained over 41 datasets showed how OCEAn is a good option to deal with ordinal classification problems. In terms of MAE, our proposal improved results in 34 of 41 datasets; in terms of MZE, it improved results in 32 cases, representing improvement ratios of 82.92 % and 78.04 % of cases, respectively.

In future work, we propose adaptating the OCEAn algorithm to current leading technologies, such as data streaming and Big Data paradigms, which requires a thorough examination of the effectiveness and scalability possibilities of our proposal. We would like to consider other parameters for the optimization of our proposal, such as the cost matrix, because we believe it could be a good option to conduct experiments with different types of matrices, due to the diversity of the datasets. Finally, given that we have not done any preprocessing on the datasets, we will explore preprocessing techniques in future approaches of this problem, such as feature or instance selection, which might improve the final classification result, for which we plan to use the ideas behind active learning, as mentioned in [2,23].

CRediT authorship contribution statement

Belén Vega-Márquez: Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft, Visualization. **Isabel A. Nepomuceno-Chamorro:** Validation, Investigation, Resources, Writing - review & editing, Supervision, Project administration. **Cristina Rubio-Escudero:** Validation, Investigation, Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition. **José C. Riquelme:** Conceptualization, Methodology, Validation, Formal analysis, Resources, Supervision, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 14
MAE comparison with base classifiers. For each dataset, the MAE value obtained with each of the 18 base classifiers is shown, as well as the average (column "Average") of these 18 values and the final value obtained by our proposal (column "OCEAn").

	MAE																			
	NaivesBayes	MLP	SMO	IBk	KStar	AdB	Bag	LB	J48	DS	LMT	RF	REPTree	PART	JRip	Log	CVR	BN	Average	OCEAn
CL	0.433	0.233	0.533	0.367	0.333	0.433	0.333	0.500	0.333	0.333	0.467	0.267	0.467	0.333	0.633	0.600	0.333	0.367	0.405	**0.303**
PA	0.267	0.311	0.311	0.444	0.467	0.267	0.333	0.356	0.289	0.333	0.289	0.222	0.333	0.356	0.511	0.356	0.267	0.200	0.328	**0.268**
SS	0.415	0.508	0.431	0.415	0.338	0.369	0.615	0.354	0.385	0.354	0.462	0.415	0.569	0.354	0.415	0.569	0.508	0.431	0.439	**0.373**
SU	0.262	0.277	0.292	0.400	0.369	0.323	0.446	0.277	0.231	0.292	0.323	0.262	0.569	0.215	0.292	0.477	0.477	0.369	0.341	**0.249**
TA	0.642	0.637	0.632	0.637	0.663	0.668	0.737	0.626	0.642	0.668	0.647	0.632	0.716	0.674	0.658	0.658	0.674	0.611	0.656	**0.637**
NT	0.022	0.037	0.104	0.030	0.052	0.052	0.056	0.030	0.096	0.222	0.022	0.033	0.074	0.085	0.063	0.033	0.041	0.033	0.060	**0.030**
BS	0.127	0.112	0.172	0.234	0.195	0.414	0.270	0.185	0.361	0.758	0.131	0.239	0.364	0.297	0.350	0.106	0.155	0.386	0.269	**0.109**
SW	0.467	0.474	0.47	0.490	0.459	0.504	0.446	0.462	0.475	0.504	0.461	0.478	0.478	0.473	0.495	0.446	0.456	0.476	0.473	**0.425**
CA	0.178	0.035	0.125	0.109	0.142	0.256	0.123	0.164	0.122	0.412	0.059	0.079	0.165	0.074	0.231	0.077	0.066	0.179	0.144	**0.038**
BO	0.787	0.613	0.533	0.600	0.960	0.747	0.520	0.653	0.600	0.733	0.587	0.533	0.600	0.653	0.680	0.640	0.560	0.453	0.636	**0.558**
TO	0.832	0.547	0.939	0.152	0.136	0.987	0.181	0.907	0.197	0.987	0.221	0.125	0.291	0.272	0.205	0.912	0.229	0.963	0.504	**0.154**
EU	0.675	0.463	0.696	0.652	0.465	0.798	0.741	0.366	0.474	0.798	0.414	0.485	0.818	0.486	0.527	0.484	0.389	0.573	0.572	**0.359**
LE	0.46	0.433	0.462	0.438	0.422	0.710	0.450	0.421	0.456	0.710	0.427	0.453	0.509	0.432	0.445	0.422	0.450	0.569	0.481	**0.412**
AU	0.638	0.435	0.577	0.431	0.585	0.700	0.665	0.365	0.385	0.700	0.408	0.350	0.819	0.435	0.508	0.696	0.469	0.554	0.540	**0.330**
WI	0.538	0.446	0.498	0.470	0.488	0.492	0.408	0.434	0.505	0.492	0.457	0.366	0.474	0.511	0.495	0.452	0.435	0.470	0.468	**0.379**
ES	0.341	0.333	0.52	0.362	0.344	0.739	0.389	0.375	0.385	0.739	0.323	0.364	0.467	0.408	0.413	0.31	0.400	0.433	0.424	**0.340**
ER	1.218	1.261	1.334	1.292	1.295	1.421	1.260	1.229	1.290	1.421	1.221	1.318	1.299	1.306	1.491	1.194	1.261	1.319	1.302	**1.220**
P5	1.017	0.933	0.958	1.000	1.033	1.067	0.983	1.033	1.283	1.067	1.092	0.925	1.108	1.092	1.233	0.958	1.150	1.017	1.052	**0.964**
M5	0.549	0.515	0.803	0.559	0.583	0.824	0.505	0.508	0.590	0.824	0.502	0.464	0.702	0.620	0.742	0.512	0.502	0.593	0.600	**0.457**
H5	0.651	0.475	0.728	0.568	0.577	0.797	0.448	0.462	0.513	0.797	0.453	0.405	0.568	0.553	0.546	0.475	0.410	0.605	0.550	**0.411**
S5	0.542	0.169	0.578	0.138	0.135	0.686	0.177	0.178	0.204	0.686	0.201	0.127	0.238	0.245	0.270	0.217	0.160	0.250	0.288	**0.122**
A5	0.889	0.698	0.889	0.907	0.811	0.916	0.732	0.748	0.870	0.916	0.697	0.717	0.799	0.900	0.971	0.679	0.689	0.882	0.817	**0.685**
B5	0.660	0.717	0.909	1.162	1.049	0.970	0.811	0.741	0.832	0.970	0.568	0.710	0.966	0.896	0.942	0.818	0.665	0.896	0.849	**0.624**
BB5	1.192	1.225	1.125	1.408	1.366	1.240	1.175	1.314	1.456	1.240	1.191	1.110	1.326	1.409	1.297	1.255	1.212	1.202	1.263	**1.138**
C5	0.561	0.591	0.769	0.641	0.802	0.904	0.644	0.640	0.657	0.904	0.590	0.560	0.840	0.737	0.936	0.711	0.689	0.743	0.717	**0.716**
CC5	1.151	1.146	1.435	1.139	1.743	1.905	1.167	1.256	1.383	1.905	1.222	1.014	1.656	1.537	1.947	1.763	1.269	1.335	**1.442**	1.612
CH5	0.978	0.731	0.997	0.963	1.047	0.966	0.814	0.848	0.966	0.966	0.725	0.809	0.919	0.959	0.999	0.682	0.738	1.003	0.895	**0.719**
CE5	0.886	0.886	0.974	1.046	0.947	1.038	0.755	0.772	0.912	1.038	0.760	0.686	0.919	0.867	0.928	0.753	0.766	0.843	0.876	**0.716**
CEE5	1.051	0.915	0.985	1.081	0.945	1.112	0.785	0.797	0.965	1.112	0.850	0.727	0.989	0.977	1.020	0.853	0.835	0.905	0.939	**0.766**
P10	2.508	2.000	2.108	2.000	1.908	2.333	2.175	2.175	2.492	2.333	2.258	2.033	2.442	2.667	2.558	2.45	2.117	2.383	2.274	**2.185**
M10	1.125	1.075	1.759	1.186	1.139	1.783	1.098	1.112	1.336	1.783	1.092	0.922	1.346	1.322	1.769	1.088	1.190	1.332	1.303	**0.989**
H10	1.505	1.187	1.516	1.271	1.159	1.685	0.953	1.019	1.195	1.685	1.076	0.866	1.332	1.262	1.507	1.190	1.018	1.222	1.258	**0.929**
S10	0.821	0.389	1.238	0.298	0.285	1.410	0.449	0.422	0.450	1.410	0.417	0.291	0.539	0.585	0.802	0.461	0.423	0.62	0.628	**0.286**
A10	1.880	1.455	1.804	1.875	1.663	1.891	1.511	1.513	1.848	1.891	1.462	1.493	1.649	1.880	1.992	1.412	1.457	1.806	1.693	**1.431**
B10	1.934	1.684	2.060	2.358	2.155	1.951	1.769	1.877	1.792	1.951	1.597	1.580	2.189	2.249	2.259	2.094	1.828	2.087	1.967	**1.600**
BB10	2.678	2.603	2.342	2.973	2.937	2.372	2.431	2.616	2.803	2.372	2.494	2.227	2.665	2.918	2.649	2.701	2.447	2.459	2.593	**2.358**
C10	1.241	1.299	1.527	1.345	1.679	1.957	1.227	1.431	1.674	1.957	1.250	1.104	1.658	1.687	1.964	1.611	1.431	1.474	1.528	**1.509**
CC10	1.151	1.146	1.435	1.139	1.743	1.905	1.167	1.256	1.383	1.905	1.222	1.014	1.656	1.537	1.947	1.763	1.269	1.335	**1.440**	1.612
CH10	1.995	1.604	2.172	1.968	2.167	2.044	1.825	1.688	2.103	2.044	1.603	1.714	2.037	2.265	2.157	1.528	1.625	2.112	1.92	**1.568**
CE10	1.797	1.819	2.039	2.198	1.925	2.055	1.513	1.611	1.875	2.055	1.617	1.417	1.895	1.875	2.076	1.633	1.595	2.032	1.834	**1.506**
CEE10	2.057	1.982	2.058	2.223	1.883	2.103	1.615	1.740	2.012	2.103	1.760	1.464	2.033	2.100	2.188	1.944	1.746	2.255	1.950	**1.615**

The best result obtained for each of the datasets is highlighted in bold type.

Table 15
MZE comparison with base classifiers. For each dataset, the MZE value obtained with each of the 18 base classifiers is shown, as well as the average (column "Average") of these 18 values and the final value obtained by our proposal (column "OCEAn").

	MZE																			
	NaivesBayes	MLP	SMO	IBk	**KStar**	AdB	Bag	LB	J48	DS	LMT	RF	REPTree	PART	JRip	Log	CVR	BN	Average	OCEAn
CL	0.367	0.133	0.300	0.333	0.267	0.367	0.300	0.300	0.200	0.333	0.333	0.200	0.367	0.200	0.367	0.367	0.200	0.267	0.288	**0.193**
PA	0.267	0.311	0.311	0.444	0.467	0.267	0.333	0.356	0.289	0.333	0.289	0.222	0.333	0.333	0.467	0.356	0.267	0.200	0.324	**0.268**
SS	0.354	0.446	0.385	0.400	0.323	0.308	0.569	0.323	0.338	0.323	0.400	0.385	0.554	0.308	0.354	0.523	0.446	0.385	0.395	**0.327**
SU	0.262	0.277	0.277	0.385	0.369	0.323	0.415	0.277	0.215	0.292	0.323	0.262	0.538	0.215	0.292	0.446	0.431	0.369	0.331	**0.249**
TA	0.526	0.479	0.500	0.468	0.489	0.637	0.611	0.505	0.511	0.637	0.495	0.484	0.632	0.537	0.605	0.495	0.511	0.495	0.534	**0.507**
NT	0.022	0.037	0.104	0.030	0.052	0.052	0.056	0.030	0.096	0.222	0.022	0.033	0.074	0.085	0.063	0.033	0.041	0.033	0.060	**0.030**
BS	0.117	0.096	0.127	0.217	0.171	0.353	0.200	0.150	0.245	0.420	0.116	0.210	0.225	0.208	0.217	0.093	0.136	0.327	0.201	**0.095**
SW	0.429	0.445	0.454	0.455	0.433	0.486	0.426	0.442	0.447	0.486	0.438	0.447	0.450	0.442	0.458	0.426	0.445	0.449	0.447	**0.416**
CA	0.157	0.031	0.124	0.092	0.116	0.219	0.108	0.144	0.106	0.299	0.056	0.070	0.140	0.062	0.178	0.073	0.060	0.160	0.121	**0.034**
BO	0.560	0.533	0.440	0.467	0.720	0.613	0.400	0.493	0.413	0.547	0.467	0.413	0.413	0.493	0.493	0.533	0.413	0.400	0.489	**0.437**
TO	0.648	0.488	0.712	0.152	0.136	0.717	0.181	0.536	0.187	0.717	0.200	0.125	0.269	0.219	0.203	0.712	0.229	0.693	0.395	**0.154**
EU	0.558	0.400	0.562	0.482	0.412	0.663	0.529	0.337	0.409	0.663	0.367	0.424	0.551	0.429	0.438	0.412	0.353	0.468	0.469	**0.337**
LE	0.427	0.402	0.433	0.399	0.390	0.606	0.414	0.390	0.419	0.606	0.398	0.406	0.454	0.399	0.408	0.392	0.419	0.515	0.437	**0.381**
AU	0.477	0.331	0.492	0.323	0.338	0.604	0.465	0.273	0.269	0.604	0.304	0.269	0.554	0.296	0.331	0.396	0.338	0.392	0.392	**0.253**
WI	0.462	0.404	0.463	0.402	0.415	0.438	0.374	0.402	0.429	0.438	0.414	0.334	0.427	0.438	0.434	0.408	0.402	0.426	0.417	**0.347**
ES	0.316	0.320	0.434	0.349	0.321	0.580	0.362	0.346	0.361	0.580	0.307	0.348	0.418	0.382	0.385	0.295	0.379	0.392	0.381	**0.319**
ER	0.741	0.734	0.783	0.754	0.756	0.774	0.764	0.742	0.738	0.774	0.744	0.760	0.762	0.754	0.799	0.732	0.765	0.774	0.758	**0.747**
P5	0.608	0.592	0.733	0.625	0.658	0.700	0.658	0.667	0.717	0.700	0.650	0.617	0.750	0.667	0.783	0.658	0.717	0.700	0.677	**0.623**
M5	0.451	0.451	0.708	0.451	0.488	0.661	0.447	0.461	0.485	0.661	0.410	0.420	0.536	0.502	0.556	0.420	0.451	0.492	0.502	**0.409**
H5	0.477	0.380	0.655	0.439	0.459	0.650	0.390	0.391	0.429	0.650	0.385	0.361	0.468	0.433	0.448	0.386	0.355	0.450	0.455	**0.355**
S5	0.395	0.158	0.565	0.138	0.135	0.600	0.174	0.171	0.195	0.600	0.186	0.127	0.229	0.221	0.222	0.197	0.158	0.241	0.261	**0.122**
A5	0.587	0.518	0.720	0.607	0.581	0.640	0.559	0.568	0.583	0.640	0.540	0.552	0.577	0.594	0.663	0.527	0.542	0.593	0.588	**0.530**
B5	0.520	0.571	0.723	0.710	0.658	0.700	0.639	0.555	0.587	0.700	0.479	0.584	0.660	0.602	0.653	0.613	0.558	0.643	0.619	**0.522**
BB5	0.726	0.733	0.772	0.770	0.768	0.783	0.785	0.759	0.764	0.783	0.732	0.775	0.788	0.762	0.782	0.746	0.756	0.794	0.765	**0.755**
C5	0.470	0.480	0.690	0.500	0.574	0.679	0.532	0.517	0.521	0.679	0.497	0.486	0.613	0.552	0.634	0.548	0.551	0.563	0.560	**0.546**
CC5	0.440	0.464	0.618	0.468	0.596	0.683	0.461	0.477	0.528	0.683	0.469	0.428	0.573	0.534	0.540	0.574	0.506	0.481	**0.529**	0.579
CH5	0.634	0.548	0.742	0.636	0.670	0.702	0.621	0.607	0.625	0.702	0.546	0.615	0.656	0.631	0.669	0.524	0.582	0.705	0.634	**0.551**
CE5	0.616	0.601	0.726	0.662	0.633	0.721	0.586	0.570	0.610	0.721	0.566	0.545	0.614	0.591	0.613	0.557	0.602	0.621	0.612	**0.547**
CEE5	0.665	0.618	0.713	0.674	0.639	0.724	0.604	0.590	0.639	0.724	0.613	0.583	0.675	0.631	0.670	0.597	0.636	0.640	0.646	**0.580**
P10	0.858	0.808	0.908	0.817	0.817	0.883	0.833	0.833	0.858	0.883	0.825	0.842	0.908	0.850	0.892	0.825	0.833	0.858	0.851	**0.838**
M10	0.641	0.647	0.895	0.681	0.685	0.827	0.658	0.658	0.698	0.827	0.675	0.607	0.736	0.712	0.763	0.658	0.722	0.708	0.711	**0.635**
H10	0.695	0.640	0.838	0.675	0.643	0.827	0.593	0.606	0.652	0.827	0.616	0.575	0.701	0.649	0.702	0.635	0.616	0.649	0.674	**0.575**
S10	0.507	0.330	0.750	0.276	0.269	0.800	0.392	0.381	0.368	0.800	0.357	0.276	0.422	0.410	0.479	0.383	0.374	0.479	0.447	**0.268**
A10	0.759	0.731	0.860	0.771	0.755	0.841	0.752	0.756	0.767	0.841	0.741	0.754	0.765	0.770	0.804	0.729	0.753	0.775	0.773	**0.736**
B10	0.777	0.790	0.885	0.847	0.811	0.854	0.827	0.783	0.789	0.854	0.773	0.804	0.867	0.818	0.845	0.830	0.827	0.859	0.824	**0.779**
BB10	0.878	0.872	0.892	0.885	0.883	0.881	0.892	0.867	0.873	0.881	0.871	0.886	0.900	0.880	0.888	0.876	0.878	0.896	0.882	**0.876**
C10	0.678	0.697	0.839	0.687	0.751	0.843	0.712	0.724	0.747	0.843	0.720	0.680	0.775	0.747	0.808	0.752	0.774	0.760	**0.752**	0.759
CC10	0.649	0.654	0.831	0.651	0.749	0.841	0.694	0.687	0.688	0.841	0.674	0.646	0.770	0.725	0.791	0.75	0.742	0.725	**0.728**	0.787
CH10	0.795	0.761	0.884	0.79	0.823	0.855	0.821	0.768	0.802	0.855	0.775	0.801	0.838	0.816	0.846	0.742	0.810	0.867	0.813	**0.759**
CE10	0.769	0.789	0.873	0.828	0.807	0.858	0.780	0.773	0.792	0.858	0.767	0.763	0.823	0.790	0.827	0.757	0.798	0.852	0.805	**0.759**
CEE10	0.807	0.806	0.861	0.828	0.808	0.864	0.799	0.795	0.812	0.864	0.795	0.780	0.837	0.817	0.843	0.799	0.816	0.880	0.822	**0.788**

The best result obtained for each of the datasets is highlighted in bold type.

Acknowledgment

This research has been funded by FEDER/Ministerio de Ciencia, Innovación y Universidades - Agencia Estatal de Investigación/Proyecto TIN2017-88209-C2 and by the Andalusian Regional Government under the project BIDASGRI: Big Data technologies for Smart Grids (US-1263341).

Appendix A

The following tables show the effectiveness of the individual base algorithms compared to our proposal. There is a column for each of the algorithms used as base classifiers, and a column (“Average”) indicating what is the average, for each dataset, of MAE and MZE, respectively, among all base classifiers that compose OCEAn. If the average is compared to the results obtained by OCEAn, we obtained better results in 39 out of 41 datasets in terms of MAE and 38 out of 41 datasets in terms of MZE.

For each of the rows, the best performing algorithm is shown in bold. In both tables, it can be seen that in certain cases, some individual classifiers outperform our proposal. However, as we obtained better results on average, our proposal eliminates the decision of which algorithm to choose, which represents a significant advantage (see Tables 14 and 15).

References

[1] A. Alsaeedi, M.Z. Khan, Software defect prediction using supervised machine learning and ensemble techniques: a comparative study, Journal of Software Engineering and Applications 12 (2019) 85–100.
[2] S. Begum, S. Bera, D. Chakraborty, R. Sarkar, Breast cancer detection using feature selection and active learning, in: Computer, communication and electrical technology: proceedings of the international conference on advancement of computer communication and electrical technology (ACCET 2016), West Bengal, India, 21–22 October 2016, The British Institute of Radiology, 2017, pp. 43–48..
[3] A. Benavoli, G. Corani, F. Mangili, Should we really use post-hoc tests based on mean-ranks?, The Journal of Machine Learning Research 17 (2016) 152–161
[4] D. Brodic, Methodology for the evaluation of the algorithms for text line segmentation based on extended binary classification, Measurement Science Review 11 (2011) 71–78.
[5] J.S. Cardoso, R. Sousa, I. Domingues, Ordinal data classification using kernel discriminant analysis: A comparison of three approaches, in: 2012 11th International Conference on Machine Learning and Applications, IEEE, 2012, pp. 473–477.
[6] K. Chang, C. Chen, Y. Hung, Ordinal hyperplanes ranker with cost sensitivities for age estimation, in: CVPR 2011, 2011, pp. 585–592.
[7] W. Chu, Z. Ghahramani, Gaussian processes for ordinal regression, Journal of Machine Learning Research 6 (2005) 1019–1041.
[8] W. Chu, Z. Ghahramani, F. Falciani, D.L. Wild, Biomarker discovery in microarray gene expression data with gaussian processes, Bioinformatics 21 (2005) 3385–3393.
[9] W. Chu, S.S. Keerthi, Support vector ordinal regression, Neural Computation 19 (2007) 792–815.
[10] K. Crammer, Y. Singer, Online ranking by projecting, Neural Computation 17 (2005) 145–175.
[11] S. Das, D. Venugopal, S. Shiva, A holistic approach for detecting ddos attacks by using ensemble unsupervised machine learning, Future of Information and Communication Conference, Springer. (2020) 721–738.
[12] D. Dua, C. Graff, UCI machine learning repository, 2017, http://archive.ics.uci.edu/ml..
[13] E. Frank, M. Hall, A simple approach to ordinal classification, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2001, doi:10.1007/3-540-44795-4_13..
[14] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes, Pattern Recognition 44 (2011) 1761–1776.
[15] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power, Information Sciences 180 (2010) 2044–2064.
[16] S. Garcia, F. Herrera, An extension on “statistical comparisons of classifiers over multiple data sets” for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.
[17] L. Gaudette, N. Japkowicz, Evaluation methods for ordinal classification, in: Canadian Conference on Artificial Intelligence, Springer, 2009, pp. 207–210.
[18] A.L. Greil, K.S. Slauson-Blevins, M.H. Lowry, J. McQuillan, Concerns about treatment for infertility in a probability-based sample of us women, Journal of Reproductive and Infant Psychology 38 (2020) 16–24. https://doi.org/10.1080/02646838.2019.1587395, 10.1080/02646838.2019.1587395, arXiv: https://doi.org/10.1080/02646838.2019.1587395. pMID: 30892066..
[19] P.A. Gutierrez, M. Perez-Ortiz, J. Sanchez-Monedero, F. Fernandez-Navarro, C. Hervas-Martinez, Ordinal regression methods: survey and experimental study, IEEE Transactions on Knowledge and Data Engineering 28 (2015) 127–146.
[20] K. Hechenbichler, K. Schliep, Weighted k-nearest-neighbor techniques and ordinal classification, 2004..
[21] J.H. Holland, Genetic algorithms, Scientific American 267 (1992) 66–73.
[22] C.W. Hsu, C.J. Lin, A comparison of methods for multiclass support vector machines, IEEE Transactions on Neural Networks 10 (1109/72) (2002) 991427.
[23] R. Ibrahim, N.A. Yousri, M.A. Ismail, N.M. El-Makky, Multi-level gene/mirna feature selection using deep belief nets and active learning, in: 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2014, pp. 3957–3960.
[24] T. Kato, H. Kashima, M. Sugiyama, K. Asai, Multi-task learning via conic programming, Advances in Neural Information Processing Systems 20 (2007) 737–744.
[25] W. Kotlowski, R. Slowinski, On nonparametric ordinal classification with monotonicity constraints, IEEE Transactions on Knowledge and Data Engineering 25 (2012) 2576–2589.
[26] H. Liu, M. Cocea, Nature-inspired framework of ensemble learning for collaborative classification in granular computing context, Granular Computing 4 (2019) 715–724.
[27] J.M. Luna-Romera, M. Martínez-Ballesteros, J. García-Gutiérrez, J.C. Riquelme, External clustering validity index based on chi-squared statistical test, Information Sciences 487 (2019) 1–17.
[28] M. Lázaro, F. Herrera, A.R. Figueiras-Vidal, Ensembles of cost-diverse bayesian neural learners for imbalanced binary classification, Information Sciences 520 (2020) 31–45, https://doi.org/10.1016/j.ins.2019.12.050, http://www.sciencedirect.com/science/article/pii/S0020025519311612.
[29] Z. Niu, M. Zhou, L. Wang, X. Gao, G. Hua, Ordinal regression with multiple output cnn for age estimation, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[30] D. Opitz, R. Maclin, Popular ensemble methods: An empirical study, Journal of Artificial Intelligence Research 11 (1999) 169–198.

[31] J. de Oña, R. de Oña, F.J. Calvo, A classification tree approach to identify key factors of transit service quality, Expert Systems with Applications 39 (2012) 11164–11171, https://doi.org/10.1016/j.eswa.2012.03.037, http://www.sciencedirect.com/science/article/pii/S095741741200560X.
[32] J.A. Parejo Maestre, J. García, A. Ruiz Cortés, J.C. Riquelme Santos, Statservice: Herramienta de análisis estadístico como soporte para la investigación con metaheurísticas, in: VIII Congreso Español Sobre Metaheurísticas, Algoritmos Evolutivos y Bio-Inspirados, 2012, Albacete, España, 2012..
[33] PASCAL, Pascal (Pattern Analysis, Statistical Modelling and Computational Learning) machine learning benchmarks repository, 2011, http://mldata.org/..
[34] W. Pedrycz, Information granules and their use in schemes of knowledge management, Scientia Iranica 18 (2011) 602–610.
[35] R. Potharst, J.C. Bioch, A decision tree algorithm for ordinal classification, in: D.J. Hand, J.N. Kok, M.R. Berthold (Eds.), Advances in Intelligent Data Analysis, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999, pp. 187–198.
[36] N. Rodríguez-Barroso, A.R. Moya, J.A. Fernández, E. Romero, E. Martínez-Cámara, F. Herrera, Deep learning hyper-parameter tuning for sentiment analysis in twitter based on evolutionary algorithms, in: 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), 2019, pp. 255–264, https://doi.org/10.15439/2019F183.
[37] O. Sagi, L. Rokach, Ensemble learning: A survey. WIREs Data Mining and Knowledge Discovery 8 (2018) e1249. https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249, 10.1002/widm.1249, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1249..
[38] M.P. Sesmero, J.M. Alonso-Weber, A. Sanchis, Cce: An ensemble architecture based on coupled ann for solving multiclass problems, Information Fusion 58 (2020) 132–152, https://doi.org/10.1016/j.inffus.2019.12.015, http://www.sciencedirect.com/science/article/pii/S1566253519305469.
[39] S. Sun, Traffic flow forecasting based on multitask ensemble learning, in: Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 2009, pp. 961–964..
[40] M.K. Tomczyk, M. Kadziński, Emosor: Evolutionary multiple objective optimization guided by interactive stochastic ordinal regression, Computers & Operations Research 108 (2019) 134–154, https://doi.org/10.1016/j.cor.2019.04.008, https://www.sciencedirect.com/science/article/pii/S0305054819300917.
[41] V.M. Vargas, P.A. Gutiérrez, C. Hervás-Martínez, Cumulative link models for deep ordinal classification, Neurocomputing (2020).
[42] A. Verma, Evaluation of classification algorithms with solutions to class imbalance problem on bank marketing dataset using weka, International Research Journal of Engineering and Technology 5 (2019) 54–60.
[43] W. Waegeman, L. Boullart, An ensemble of Weighted Support Vector Machines for Ordinal Regression, International Journal of Electrical and Electronics Engineering (2009).
[44] Q. Wang, L. Zhang, Ensemble learning based on multi-task class labels, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2010, pp. 464–475.
[45] Q. Wang, L. Zhang, M. Chi, J. Guo, Mtforest: Ensemle decision trees based on multi-task learning, in: ECAI, 2008, pp. 122–126.
[46] I.H. Witten, E. Frank, Data mining: practical machine learning tools and techniques with java implementations, Acm Sigmod Record 31 (2002) 76–77.
[47] L. Xu, S. Ding, A novel clustering ensemble model based on granular computing, Applied Intelligence (2021) 1–15.
[48] Y. Yao, Artificial Intelligence Perspectives on Granular Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 17–34, https://doi.org/10.1007/978-3-642-19820-5_2.
[49] P. Yildirim, U.K. Birant, D. Birant, M.H. Moghaddam, EBOC: Ensemble-Based Ordinal Classification in Transportation, Journal of Advanced Transportation (2019), https://doi.org/10.1155/2019/7482138.
[50] Y. Zhang, H. Zhang, J. Cai, B. Yang, A weighted voting classifier based on differential evolution, in: Abstract and Applied Analysis, Hindawi, 2014.

Belén Vega received a B.S. degree in health engineering in 2018 and an M.Sc. degree in computer science in 2019 from the University of Sevilla, Spain, where she is currently pursuing a Ph.D. in computer sciences and artificial intelligence in the Department of Computing Systems and Languages. Her current interests are the use of data science techniques such as artificial intelligence and data mining for its application in health and nutrition.

Isabel A. Nepomuceno-Chamorro received her Ph.D. in applied computer science from the University Pablo de Olavide, Spain in 2011. She is a tenured lecturer in the Department of Computing Systems and Languages, University of Sevilla, and a researcher in the Minerva Lab from the University of Seville. Her primary research interests include machine learning techniques and their application in the area of personalized medicine.

Cristina Rubio-Escudero received her Ph.D. in computer science from the University of Granada, Spain in 2007. She is a tenured lecturer in the Department of Computing Systems and Languages, University of Sevilla, and a researcher in the Data Science & Big Data research lab. She has directed some regional and national research. She is co-author of more than 50 articles, book chapters, and conference papers. Her research interests include data science in general and clinical and environmental data mining in particular.

José C. Riquelme received an M.Sc. degree in mathematics and a Ph.D. degree in computer science from the University of Seville, Spain. Since 1987, he has been with the Department of Computer Science, University of Seville, where he is currently a full professor. His primary research interests include data mining, machine learning techniques, and evolutionary computation.