Neural Computing and Applications (2023) 35:5335–5349
https://doi.org/10.1007/s00521-022-07974-z

ORIGINAL ARTICLE

Unsupervised instance selection via conjectural hyperrectangles

Fatih Aydin[1]

Received: 10 June 2022 / Accepted: 17 October 2022 / Published online: 2 November 2022

Abstract
Machine learning algorithms spend a lot of time processing data because they are not fast enough to commit huge data sets. Instance selection algorithms especially aim to tackle this trouble. However, even instance selection algorithms can suffer from it. We propose a new unsupervised instance selection algorithm based on conjectural hyper-rectangles. In this study, the proposed algorithm is compared with one conventional and four state-of-the-art instance selection algorithms by using fifty-five data sets from different domains. The experimental results demonstrate the supremacy of the proposed algorithm in terms of classification accuracy, reduction rate, and running time. The time and space complexities of the proposed algorithm are log-linear and linear, respectively. Furthermore, the proposed algorithm can obtain better results with an accuracy-reduction trade-off without decreasing reduction rates extremely. The source code of the proposed algorithm and the data sets are available at https://github.com/fatihaydin1/NIS for computational reproducibility.

Keywords Machine learning · Instance reduction · Prototype selection · Big data

1 Introduction

Machine Learning is a study field, which aims to bring in learning talent for automata to discover patterns in real-world problems. Two of the machine learning types are supervised and unsupervised learning, which are commonly used in lots of problems. Given a training set $T = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in \mathbb{R}^{m \times d}$, which contains m instances and is described by d features and c classes, from input points $X = \{x_i\}_{i=1}^{m} \Rightarrow x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(d)}\right) \in \mathbb{R}^d$, $i = 1, \ldots, m$ and labels $y_j \in Y, j = 1, \ldots, c$. A supervised learning algorithm seeks a function $h : X \rightarrow Y$ that is an element of the hypothesis space H. As for unsupervised learning, let D be an unknown distribution function. Accordingly, unsupervised learning aims to learn $X \sim D$ concerning $h \in H$. While applying supervised and unsupervised learning tasks, batch learning is preferred in the industry because it is faster and easier to implement in comparison to online learning, which takes a real-time stream of data. Further, both learning types suffer from big data. Huge datasets are widespread in lots of areas, including data mining, text categorization, financial forecasting, multimedia databases, genome sequences, and meteorological, financial, industrial, and science repositories. Undoubtedly, the amount of data is quickly growing in all domains. Hence, constructing a model would be more costly and requires more clusters. Furthermore, outliers and noises participated in the data set while collecting more data tend to impact the performance more. To address these troubles, data reduction methods are often employed. Thus, outliers and noises are discarded, and the data set is refined. Data reduction could be conducted by feature selection or instance selection algorithms, or both. Some strategies used in feature selection are low variance filers, principal component analysis, high correlation filters, forward feature construction, and backward feature elimination methods [1]. In this paper, we handle instance selection for data reduction in supervised and unsupervised learning problems. In this regard, the objective in instance selection is to find an optimal subset of the input set, i.e., $S \subset X$. In this respect, the use of the instance selection (i.e., prototype selection) algorithms would be effective in building models rapidly.

Instance selection is a process to discard noisy or redundant data [2]. In other words, the objective of the

✉ Fatih Aydin
fatih.aydin@balikesir.edu.tr

[1] Department of Computer Engineering, Faculty of Engineering, Balikesir University, Balikesir, Turkey

instance selection algorithms is to remove useless data from the training set. In consequence, the classification accuracies of the selected subset and original data set are either close to or the almost same as each other. Instance selection would be useful for decreasing the training and test time for lazy learners (e.g., k-Nearest Neighbors [KNN]) and high computational costly algorithms such as Support Vector Machine (SVM) and Neural Networks. In addition, instance reduction algorithms are presented to tackle problems in the fields such as time series, class-imbalanced data sets, distributed learning, noise sensitiveness, monotonic data sets, and lazy learners. In the literature, the approaches used in developing instance selection algorithms are the nearest neighbor, evolutionary computation, meta-learning, artificial immune model, divide-and-conquer strategy, Bayesian approach, cluster-based approach, hashing approach, geometry-based approach, instance ranking approach, density-based approach, the edge detection approach in image processing, metaheuristic algorithms, swarm intelligence algorithms, and graph-based approach. There are a few common properties in instance selection algorithms: Evaluation of search, type of selection, and direction of search [2–4]. Besides, there are four criteria to contrast instance selection algorithms: Storage requirement, noise resistance, classification accuracy, and running time [3].

All these approaches have a variety of advantages and disadvantages. Besides, we think that instance selection methods would persist to be incomplete due to the accuracy-reduction dilemma and no single approach can overwhelm the other approaches over all the data sets. In other words, there exist data sets where each method is successful and unsuccessful because of their assumptions and characteristics. Despite this, we can compare instance selection algorithms in terms of criteria such as reduction rate, classification accuracy, and running time. In this paper, we try to focus on a faster instance selection and a simple accuracy rate-reduction ratio trade-off by using a single parameter. This is because instance selection algorithms suffer from big data just like classification, regression, and clustering algorithms, as well. Moreover, adjusting the accuracy rate—reduction ratio trade-off is a serious problem in instance selection algorithms that have lots of meta parameters and this situation complicates the management of the process. To efficiently implement and deploy instance selection algorithms in terms of software/hardware, it is a significant matter that the algorithm is also comprehensible and easily applicable. In this study, as devising an algorithm corresponding to the aforementioned criteria, conjectural hyper-rectangles are employed. Hyperrectangle, whose edges are all mutually perpendicular in Euclidean n-space E^n is the generalization of a rectangle to higher dimensions. We call defining the pseudo-hyper-rectangles the conjectural hyper-rectangles instead of real hyper-rectangles to contain points in E^n. Hyper-rectangles are used in performing various tasks in the machine learning area (e.g., classification, clustering, instance selection, etc.). Coming up with a solution to a problem by creating hyper-rectangles in E^n is more efficient than using other geometric structures and easier than forming them. Therefore, using those is preferable to others. But it takes a long time on data sets with huge volumes to calculate the proper positions of real hyper-rectangles. Particularly, we remind that hyper-rectangles can intersect with each other. In this case, many unnecessary hyper-rectangles have been created. In this respect, we have preferred to exploit conjectural hyper-rectangles both not to describe unnecessary hyper-rectangles and to speed up the algorithm by not creating real hyper-rectangles. Lastly, we develop an instance selection algorithm to use in both supervised and unsupervised learning problems.

In this paper, we propose a fast unsupervised instance selection algorithm based on conjectural hyper-rectangles. The time complexity of our proposed method is log-linear. Besides, the proposed algorithm has obtained remarkable results on the data sets used in the experiments. The main contributions of the proposed method are as follows:

- The proposed unsupervised algorithm is straightforward and rapidly processes huge data sets.
- The accuracy-reduction trade-off can be easily adjusted by the scaling rate parameter, which is a single parameter of the proposed algorithm.

The rest of the paper is arranged in the following. In Sect. 2, we introduce our algorithm. Section 3 presents a literature review. In Sect. 4, we introduce the experimental setup. In Sect. 5, we share results and discussion in detail. Lastly, we finalize in Sect. 6.

2 Background

Moving backward in time, it is seen that one of the first algorithms that have been developed to reduce the data set is Condensed Nearest Neighbor (CNN) [5]. CNN is an iterative algorithm and starts with an empty subset. In the next step, CNN randomly picks up an instance from the training data and adds it to the subset if it is misclassified when using the subset as training data. The stop criterion is that there remain no more prototypes. CNN does not guarantee finding the optimum subset. Further, it generates different subsets on every run due to random instance selection [6]. The Reduced Nearest Neighbor (RNN) has been introduced as a modification to CNN to obtain more minimal subsets [7]. Ullmann developed an algorithm that obtains smoother boundaries by improving CNN and RNN

[8]. Selective Nearest Neighbor (SNN) has been presented as an extension of CNN. SNN guarantees that the nearest neighbor of each instance of the original data set is a member of the subset and a minimal subset. But SNN is a difficult and complex algorithm to implement [9]. Tomek proposed two methods to improve CNN and claimed that the methods ensure a minimal subset and a boundary close to the decision boundary [10]. Gowda and Krishna proposed a method to improve CNN. The proposed method precludes the holding of inner instances in the condensed set, trying to add near instances to the decision boundary [11]. IB2 and IB3 methods have been introduced to discard fluctuated instances from the training set [12]. Fast Condensed Nearest Neighbor Family (FCNN) contains a set of algorithms that run on big data fast. The time complexity of the algorithms is quadratic in the worst case [13]. One of the first methods that focus to remove noisy instances is Edited Nearest Neighbor (ENN). ENN aims to discard noisy instances in a training set [14]. The Edition Method Based on Local Sets (ELS) has been proposed as a new parameter-free algorithm based on local sets with natural neighbors to obtain more acceptable boundaries and to filter noisy instances efficiently [15].

The algorithms in the literature are categorized into several approaches in terms of the applied techniques: various hybrid methods that combine condensing and editing approaches have been proposed [16–18]. As for ensemble approaches, a variety of ensemble IS methods are proposed [6, 19, 20]. In respect of the use of evolutionary approaches, many methods have been proposed [21–23]. For the use of the computational methods, the MapReduce solution for Prototype Reduction (MRPR) has been introduced as a new instance selection algorithm to accelerate the classification stage and decrease the storage needs and the noise sensitiveness of the nearest neighbor rule [24]. In the matter of hashing, some approaches have been proposed [25–27]. To speed up SVM, Shell Extraction (SE) has been proposed [28] and in addition, two instance selection methods, based on a nature-inspired metaheuristic algorithm and a normal individual-based swarm intelligence algorithm, have been proposed [29]. Relating to scoring the instances, the instance selection algorithms based on the ranking approach have been proposed [30, 31]. Concerning the solution approaches to different data set types (e.g., class-imbalanced data sets), a variety of methods have been proposed [32, 33]. Finally, the point of geometrical approaches, many methods have been proposed [34–38].

Judging the works accomplished recently, it is seen that serious efforts are made to solve important problems. The major hardships in GA-based instance selections are high computational complexity and decreasing performance with the dataset size growth. To tackle the related problems in a three-step strategy, fuzzy clustering decomposition based on a genetic algorithm has been proposed for regression problems [39]. A generic cluster-oriented instance selection algorithm that conducts an unsupervised K-Means Clustering algorithm on the training set and with a given selection rate and chooses instances from the centers and the boundaries of the clusters has been proposed for classification problems [1]. Intrusion detection data sets are represented by huge data, which affects the learning of the classifier. Hence, there exists a requirement to decrease the size of the data sets. Therefore, a new fast instance selection method is proposed to provide better efficiency during the training stage, without greatly impacting the effectiveness of the intrusion detection scheme [40]. The preservation rough set model based on rough sets has been proposed, which deals with the instance selection problem to enhance lazy learners in hybrid and incomplete data sets [41]. A method that is inspired by the cross-validation and divide-and-conquer strategies and uses genetic algorithms and an open-source framework to choose an optimum instance subset from huge data has been proposed, which relies on combined information entropy [42].

3 The proposed algorithm

The proposed algorithm discards instances by exploiting hyper-rectangles. Essentially, since detecting directly hyper-rectangles can slow down the running time of the instance selection process, we do not try to find hyper-rectangles directly for solving the trouble. Instead, we utilize the concept of data scaling in Statistics. Thus, we can avoid the cost of calculating the positions of real hyper-rectangles and indirectly form hyper-rectangles. We call hyper-rectangles created by adopting this approach conjectural hyper-rectangles. Figure 1 shows the positions of the conjectural hyper-rectangles on the original seeds data set. Accordingly, thirty conjectural hyper-rectangles have been constituted by the proposed method on the original data set. These hyper-rectangles are rectangles in two-dimensional space. Accordingly, the proposed method randomly keeps only a single instance in each rectangular region and disposes of the rest of the instances from their own rectangular region. For high dimensional space, these rectangular regions are hyper-rectangular, and the approach of the proposed method is the same way for hyper-rectangular regions, as well. The proposed method relies on data scaling in Statistics to calculate conjectural hyper-rectangles. In this respect, the proposed method assumes that data is normally distributed. Considering one-dimensional space, when we scale a number of data by their standard deviation and then, when we round the scaled

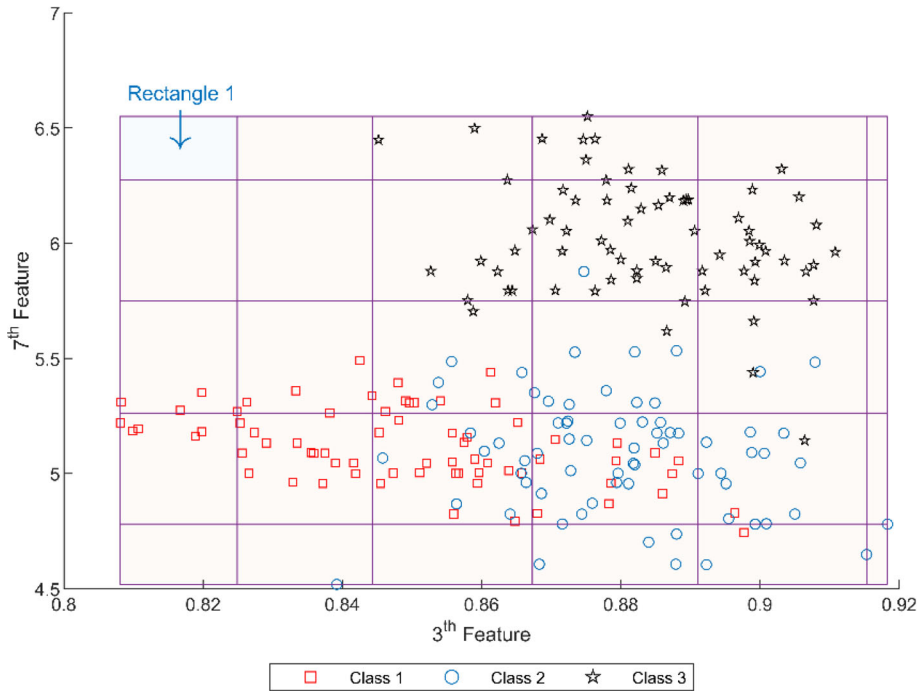

Fig. 1 The positions of the conjectural hyper-rectangles on the original seeds data set

values by a certain rule some values would be in the same range. These ranges are regarded as the unevenly spaced points of a number line in one-dimensional space. For two-dimensional space, each number line corresponds to an axis such as the x-axis and y-axis, and these axes constitute a cartesian plane. As a result of intersections of points on each axis, a grid emerges, which is composed of rectangles. For high-dimensional spaces, there exists a hyper-grid, which consists of hyper-rectangles. In a data space, data takes a place in these hyper-rectangles, and the goal of the proposed method is to retain only the first data point in these hyper-rectangles and discard the rest of the data points in each hyper-rectangle. The proposed method is simply constructed in this way.

Given a training set $T = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in \mathbb{R}^{m \times d}$, which contains m instances and is described by d features and c classes, from training points $X = \{x_i\}_{i=1}^{m} \Rightarrow x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(d)}\right) \in \mathbb{R}^d$, $i = 1, \ldots, m$ and labels $y_j \in Y$, $j = 1, \ldots, c$. The range of a data $x_i^{(k)}$ in any dimension, $k = 1, \ldots, d$, is calculated as in Eq. (1). Thus, we calculate in which range the data within each dimension (or feature) is. In other words, we scale all values. Eventually, the real positions of the conjectural hyper-rectangles are implicitly determined through $pos^{(k)} = x'\sigma_{x^{(k)}} + \min\left(x^{(k)}\right)$ from Eq. (1). This is useful to view where the locations of the conjectural hyper-rectangles are on the original data set.

$$x' = \left\lceil \frac{x_i^{(k)} - \min\left(x^{(k)}\right)}{\sigma_{x^{(k)}}} \right\rceil \tag{1}$$

The proposed method assumes that data is normally distributed. In this regard, as values in Normal Distribution move away with a certain deviation from the mean, the probabilities that they are within the related range also change. Hence, all the hyper-rectangles are not the same size. Accordingly, the maximum and minimum values of x' in Eq. (1) are six and zero, respectively, according to the three-sigma rule of thumb, for example. In this case, each data cell can take seven possible values. In other words, it has seven possible states. But we note that data distribution can be heavy-tailed or light-tailed relative to a normal distribution or not symmetric. We add a scaling rate α to Eq. (1) as shown in Eq. (2) to adjust the size of hyper-rectangles. As the value of the scaling rate decreases, the size of the hyper-rectangles increases, and accordingly the number of hyper-rectangles decreases, and vice versa. Further, the default value of the scaling rate is 1.

$$x' = \left\lceil \alpha \left(\frac{x_i^{(k)} - \min\left(x^{(k)}\right)}{\sigma_{x^{(k)}}} \right) \right\rceil \tag{2}$$

The proposed algorithm runs depending on the scaling rate to reduce instances. We call the proposed instance reduction algorithm the Nimble Instance Selection (NIS) and describe it in Algorithm 1.

Algorithm 1: Nimble Instance Selection (NIS)

Input:

$\mathbf{X} = \{\boldsymbol{x_i}\}_{i=1}^{m} \Longrightarrow \boldsymbol{x_i} = \left(\boldsymbol{x_i^{(1)}}, \boldsymbol{x_i^{(2)}}, \ldots, \boldsymbol{x_i^{(d)}}\right)$: Training set

$\boldsymbol{\alpha}$: The scaling rate (by default, 1)

Output:

I: The indices of the selected instances

1: **S** ← Score **X** by Eq. (2)

2: **I** ← Keep the index of just the first one of the same instances from **S**

Now, we calculate the time and space complexities of the algorithm. NIS measures the rounded distance of the instances from the minimum value in terms of the standard deviation and accordingly removes some repetitive instances. The upper bound time and space complexities of scoring the instances are $O\left(2md\right)$ and $O\left(md\right)$, respectively. The upper bound time and space complexities of searching unique instances are $O\left(md\log_2 md + md\right)$ and $O\left(md\right)$, respectively. In consequence, the total time complexity is $O\left(md\log_2 md\right)$ and the total space complexity is $O\left(md\right)$.

Now let us find out the equation that gives the average number of reduced instances by making various assumptions. To facilitate calculations, we assume that all the dimensions have the same characteristics. The probability of successful events in a sample of size n that are selected with replacement from a data set of m instances is shown in Eq. (3).

$$b(n,\ m,\ p) = \binom{m}{n} p^n q^{(m-n)} \tag{3}$$

where p is success probability and q is failure probability. The success probability depends on the range between data (i.e., $v\sigma - (-v\sigma) = 2v\sigma,\ v \in \mathbb{R}$ for standard normal distribution) and the number of dimensions. But we note that data distribution can be skew and kurtic relative to a normal distribution. Let $U : \Omega \rightarrow \{0,\ \cdots,\ u\},\ u \in \mathbb{Z}$ be a discrete random variable with the range values from Eq. (2) and let N be the number of total possible outcomes in U. Accordingly, the success probability is shown in Eq. (4).

$$p = \left(\frac{1}{N}\right)^d \tag{4}$$

Considering all possible cases of the same instances, we calculate the expected value of the number of selected instances (i.e., the number of reduced instances) by using Eq. (5).

$$E[Z] = \sum_{n=0}^{m} \binom{m}{n} p^n q^{(m-n)} n \tag{5}$$

where, $Z : \Omega \rightarrow \{1, \cdots, m\}$ is a discrete random variable with the number of the reduced instances and the symbol n refers to the number of the reduced instances. Accordingly, let us calculate the expected value of the number of selected instances by using Lemma 1.

Lemma 1 *Let* $p,\ q \in \mathbb{R}_{\geq 0} = \{x \in \mathbb{R} |\ x \geq 0\}$ *and* $m,\ n \in \mathbb{Z}_{\geq 0} = \{x \in \mathbb{Z} |\ x \geq 0\}$ *such that*

$$mp(p+q)^{m-1} = \sum_{n=0}^{m} \binom{m}{n} p^n q^{(m-n)} n$$

.

Since $E[Z] = \sum_{n=0}^{m} \binom{m}{n} p^n q^{(m-n)} n$ from Eq. (5),

$$\sum_{n=0}^{m} \frac{m!}{n!(m-n)!} (n) p^n q^{(m-n)} \tag{6}$$

Since the n variable appears both in the numerator and in the denominator, the n variables cancel each other out. The m and p constants can be written separately from the general expression. Thus, let us rewrite Eq. (6) as in Eq. (7).

$$\sum_{n=0}^{m} (m) \frac{(m-1)!}{(n-1)!(m-n)!} p^{(n-1)} q^{(m-n)} (p) \tag{7}$$

Due to the Constant Multiple Rule of the summation, the m and p constants can be written outside the summation. Then, the upper and lower limits of the summation are properly shifted in accordance with the new expression. Accordingly, let us rearrange Eq. (7) as below.

$$mp \sum_{n=1}^{m-1} \binom{m-1}{n-1} p^{(n-1)} q^{(m-n)} \tag{8}$$

Eventually, by involving the binomial theorem $(p+q)^m = \sum_{n=0}^{m} \binom{m}{n} p^n q^{(m-n)}$, we arrive at

$$mp(p+q)^{m-1} = \sum_{n=0}^{m} \binom{m}{n} p^n q^{(m-n)} n \tag{9}$$

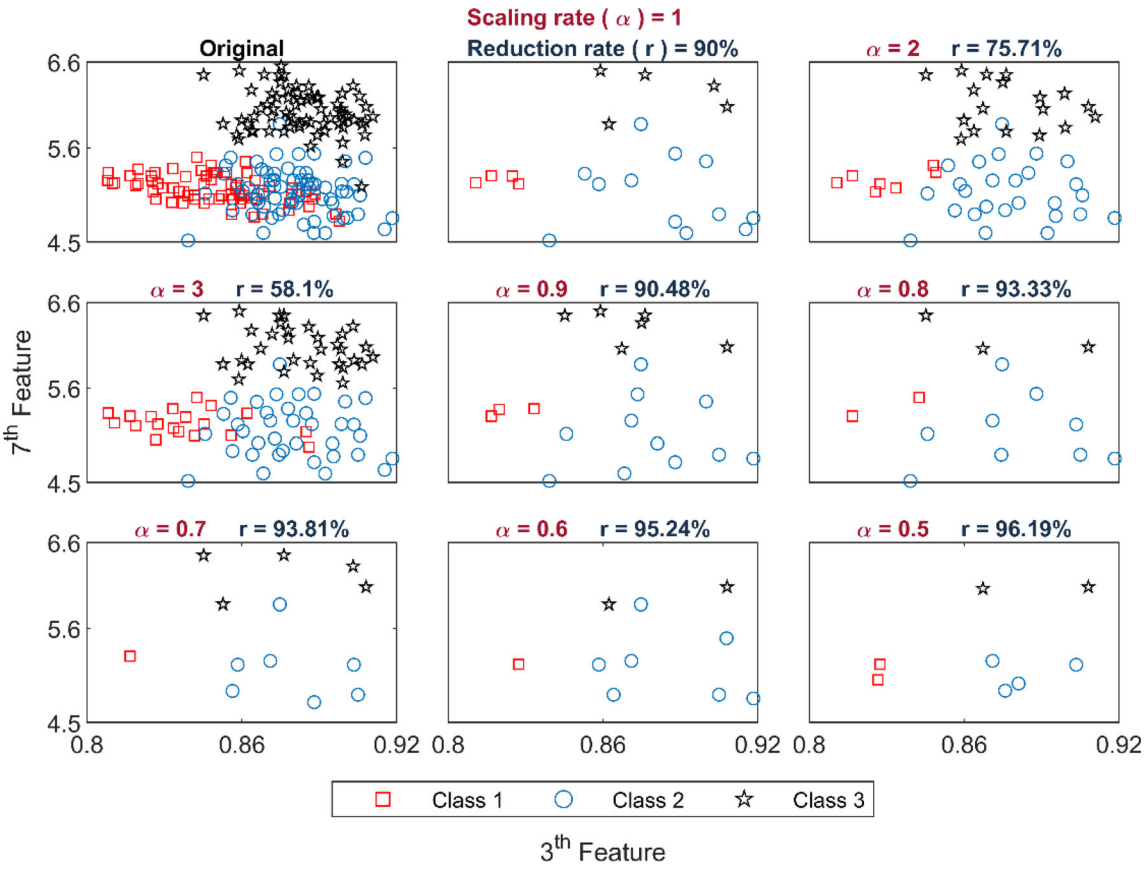

Fig. 2 The illustration of the remaining instances and reduction rates ***r*** on the seeds data set, according to the scaling rate α

We substitute Eq. (4) in Eq. (9) and since $p + q = 1$, we reach the expected value of the number of reduced instances in Eq. (10).

$$E[Z] = m\left(\frac{1}{N}\right)^d \tag{10}$$

From Eq. (10), it is concluded that large deviations and high dimensions negatively affect the expected value of the number of reduced instances. As a result, the most important factor affecting the performance of the proposed algorithm is the number of dimensions if there are no noisy data in a data set. Furthermore, since the proposed method discards at least one instance, it must satisfy $\frac{m}{N^d} \geq 1$ condition. In other words, $m \geq N^d$ condition must be satisfied. In $d > m$ or $m < N^d$ cases, the reduction rate can be increased by adjusting the scaling rate. We note that m and d are constants and α is a variable.

Figure 2 shows the instances and reduction rates r after the reduction process on the seeds data set in terms of the scaling rate α. As can be seen from the results, as the scaling rate increases the reduction rate decreases. The default value of the scaling rate is set as 1.

4 Experimental setup

In this section, we explain the experimental setup, including experimental data sets, instance selection algorithms used in the experiments, evaluation metrics, and implementations.

4.1 Data sets

We compare NIS with the state-of-the-art instance selection algorithms to measure its efficiency by using fifty-five data sets from the UCI database,[1] MATLAB sample data sets,[2] and the OpenML datasets.[3] The descriptive information belonging to those data sets is shown in Table 1.

4.2 Instance selection algorithms

We compare the proposed algorithm with the state-of-the-art instance selection algorithms. The algorithms used in the experiments are shown in Table 2. All methods are

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

[2] https://www.mathworks.com/help/stats/sample-data-sets.html.

[3] https://www.openml.org/s/88/data.

Table 1 The characteristics of the data sets used in experiments

#	Data set	Instances	Features	Classes	Imbalance ratio
1	Arrhythmia	452	279	13	122.50
2	Auditrisk	776	26	2	1.54
3	Avila	20,867	10	12	857.20
4	Banknote authentication	1372	4	2	1.27
5	Blood transfusion	748	4	2	3.20
6	Boston housing2	506	18	92	30.00
7	Breast cancer	699	9	2	1.90
8	Breast tissue	106	9	6	1.57
9	Cardiotocography3	2126	21	3	9.40
10	Cardiotocography10	2126	21	10	10.92
11	Climate model	540	18	2	10.73
12	Connectionist bench	208	60	2	1.14
13	Diabetic retinopathy	1151	19	2	1.13
14	DNA	3186	180	3	2.16
15	Ecoli	336	7	8	71.50
16	EEG_eyestate	14,980	14	2	1.23
17	Electricity	45,312	8	2	1.36
18	FisherIris	150	4	3	1.00
19	FrogsMFCCs_families	7195	22	4	65.00
20	FrogsMFCCs_genus	7195	22	8	61.03
21	FrogsMFCCs_record ID	7195	22	60	458.00
22	FrogsMFCCs_species	7195	22	10	51.15
23	Glass	214	9	6	8.44
24	Haberman	306	3	2	2.78
25	HTRU2	17,898	8	2	9.92
26	Human activity	24,075	60	5	2.34
27	Ionosphere	351	34	2	1.78
28	Leaf	340	14	30	2.00
29	Letter recognition	20,000	16	26	1.11
30	Libras movement	360	90	15	1.00
31	LSVT voice Rehabilitation	126	310	2	2.00
32	Madelon	2000	500	2	1.00
33	MAGIC gamma Telescope	19,020	10	2	1.84
34	MEU_mobile KSD	2856	71	56	1.00
35	Mozilla4	15,545	5	2	2.04
36	Nomao	34,465	118	2	2.50
37	Optical recognition	3823	64	10	1.03
38	Ovariancancer	216	4000	2	1.27
39	Page blocks	5473	10	5	175.46
40	Parkinson speech	1040	26	2	1.00
41	Poker hand	1,025,010	10	10	64,212.75
42	QSAR biodegradation	1055	41	2	1.96
43	Satellite	6435	36	6	2.44
44	Seeds	210	7	3	1.00
45	Shuttle	58,000	9	7	4558.60
46	Vehicle	846	18	4	1.10
47	Vertebral column	310	6	2	2.10
48	Vowel	990	10	11	1.00

Table 1 (continued)

#	Data set	Instances	Features	Classes	Imbalance ratio
49	WFR navigation 3	5456	2	4	6.72
50	WFR navigation 4	5456	4	4	6.72
51	WFR navigation 24	5456	24	4	6.72
52	Wine quality-red	1599	11	6	68.10
53	Wine quality-white	4898	11	7	439.60
54	Yeast	1484	8	10	92.60
55	Zoo	101	16	7	10.25

supervised and filtering-based instance selection algorithms. Besides, we run them by the default values of them in all the experiments.

4.3 Parameter setting and implementations

In this study, the baseline method means that the 1NN algorithm applies to the whole data set. Besides, we apply tenfold cross-validation to all the experiments and repeat each experiment ten times to generate the training sets with different fold combinations. The experiments have been performed in the MATLAB R2021a on an i5-8265U CPU at 1.6 GHz with 8 GB of RAM on Windows 11 Pro (64-bit). Besides, we use the default parameter values of NIS unless otherwise stated.

4.4 Evaluation criteria

We compare instance selection algorithms by using three criteria: reduction rate, classification accuracy, and running time. The lower the reduction rate, the higher the storage requirement. Furthermore, as the reduction rate increases, less decrease in classification rate is the desired situation. Finally, the algorithms are expected to process large data sets quickly. As a result, the above-mentioned criteria should be included in an algorithm.

5 Results and discussion

In this section, we conduct experiments related to the comparison of the instance selection algorithms. Figure 3 shows the results of the experiments done over the data sets in terms of classification accuracy, running time, and reduction rate. According to the results, after the instance reduction process, the highest average classification accuracy belongs to $\text{NIS}_{\text{tuned}}$ with 81.99%. The average scale rate of $\text{NIS}_{\text{tuned}}$ is 1.48 and the scale rate has been only changed for 15 out of 55 data sets. The average accuracy rate of 1NN (i.e., the baseline) on original data sets is 82.33%. The average accuracy rate of NIS on the data sets is 80.24% as the scaling rate is 1. The average accuracy rates of BPLSH, DR.LSH, LSH-IS-S, LSH-IS-F and Wilson's ENN are 81.19%, 81.45%, 81.44%, 81.69%, and 80.89%, respectively. NIS is faster than the other instance selection algorithms. Adjusting the scaling parameter of NIS does not significantly change its running time. Wilson's ENN method is the slowest instance selection algorithm because it considers the neighborhoods between data points. The average running times of NIS, $\text{NIS}_{\text{tuned}}$, BPLSH, DR.LSH, LSH-IS-S, LSH-IS-F, and Wilson's ENN are 0.03, 0.03, 342.17, 134.63, 2.30, 0.67, and 1450.50 s, respectively. The reduction rate of NIS is larger than the other instance selection methods. The average reduction rates of NIS, $\text{NIS}_{\text{tuned}}$, BPLSH, DR.LSH, LSH-IS-S, LSH-IS-F, and Wilson's ENN are 35.50%, 30.46%, 21.59%, 14.89%, 15.10%, 15.10%, and 17.65%, respectively. The selection of the fit scaling rate induces high classification accuracy and reduction rate. As a result, although NIS does not use class information it can rapidly deliver high accuracy rates and reduction rates over many data sets. Overall, the results show the reduction rate decreases as the classification accuracy of the algorithms increases. Besides, better results can be obtained by adjusting the parameter values of the algorithms. The important point is a trade-off between accuracy, reduction, and speed-up. NIS promises to be a faster algorithm and to reduce more instances than others for any data set. Furthermore, the results show that the classification accuracy can be improved more and more by changing the scaling rate parameter.

The proposed method (with default scaling parameter) cannot reduce instances on 8 out of 55 data sets. The mean range of the dimensions on the arrhythmia data set is 11.12 and the number of dimensions is 279. Accordingly, let us substitute these values in $m\left(\frac{1}{N}\right)^d = 452\left(\frac{1}{11}\right)^{279} \approx 0$. But to overcome this problem, the scaling rate parameter can be adjusted. For instance, the accuracy and reduction rates are 56.64% and 3.76%, respectively for $\alpha = 0.07$. The scaling rate parameter changes the range, i.e., N. Thus, the mean

Table 2 The state-of-the-art instance selection algorithms used in the experiments

Algorithm	Technique	Parameter
BPLSH[a] [27]	Condensation	M = 30, L = 10, W = 1
DR.LSH[b] [26]	Hybrid	M = 25, L = 10, W = 1, ST = 9
LSH-IS-S[c] [25]	Hybrid	L = 0, Y = 10, O = 4, W = 1, S = 1
LSH-IS-F[c] [25]	Hybrid	L = 0, Y = 10, O = 4, W = 1, S = 1
Wilson's ENN[d] [14]	Edit	k = 3

[a]https://github.com/mohaslani/BPLSH

[b]https://github.com/mohaslani/DR.LSH

[c]https://github.com/alvarag/LSH-IS

[d]https://github.com/LucyKuncheva/Instance_selection

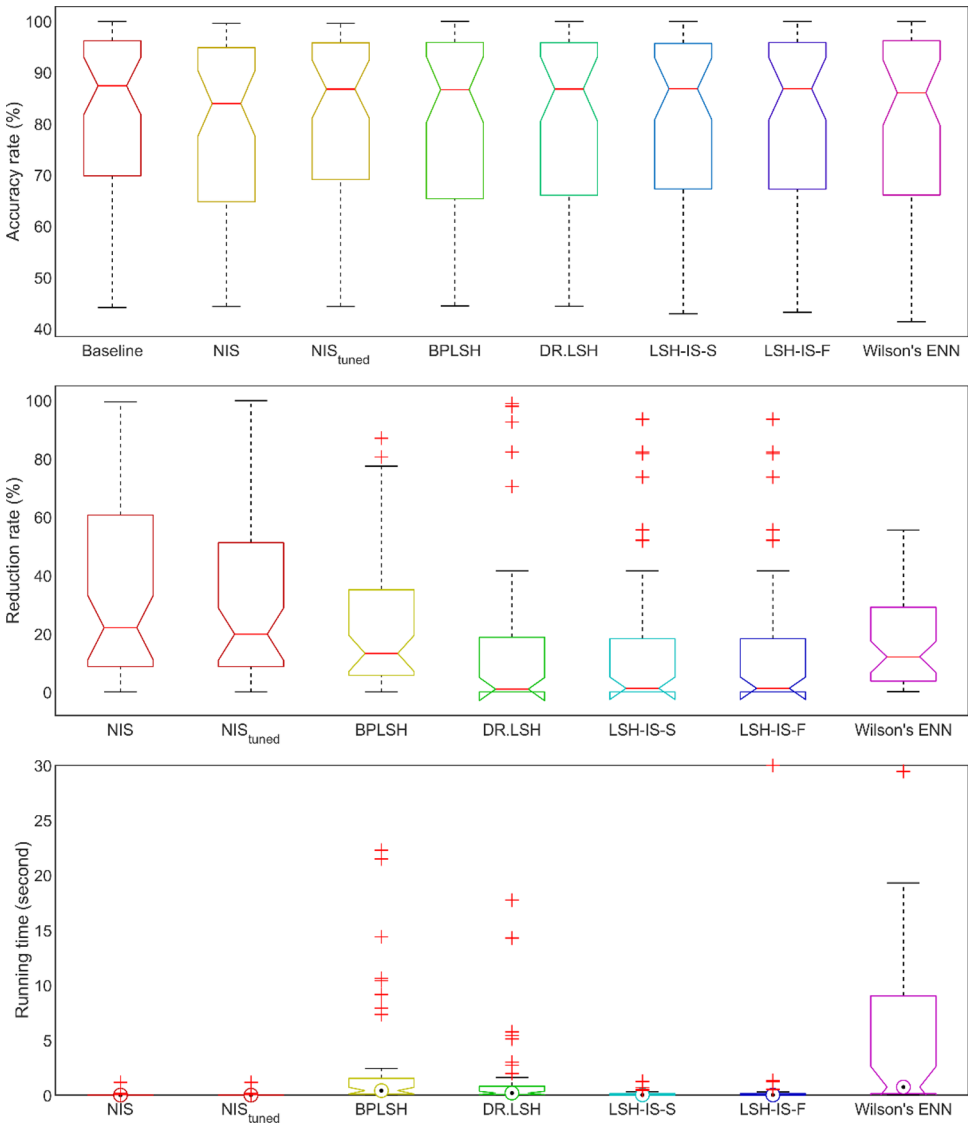

Fig. 3 The comparative results of the algorithms in terms of accuracy rate, running time, and reduction rate

range is 0.73 for $\alpha = 0.07$. As a result, tuning the scaling rate is the key to dealing with high dimensionality and large range. Besides, the mean running time of the proposed method is very close to 0. While the proposed method satisfies low running time, it also acquires high reduction rates. The summarized information about whether there is statistical importance between the results is shown in Table 3. According to Kruskal–Wallis test results,

Table 3 The mean rank results of the Kruskal–Wallis test and Friedman's test (Small values of ρ weaken the validity of the null hypothesis, which means it is not significantly different between results)

Experiment	Kruskal–Wallis test			Friedman's test		
	Accuracy rate	Running time	Reduction rate	Accuracy rate	Running time	Reduction rate
1NN	229.91	–	–	5.75	–	–
NIS	207.85	82.71	241.62	3.64	1.52	5.47
NIS_{tuned}	222.61	84.97	233.96	4.30	1.61	5.00
BPLSH	218.79	283.35	214.53	4.14	6.00	4.45
DR.LSH	221.01	258.65	149.93	5.17	5.07	2.77
LSH-IS-S	221.49	176.44	149.14	4.39	3.79	2.93
LSH-IS-F	224.16	166.84	149.14	4.98	3.26	2.93
Wilson's ENN	218.17	298.05	212.69	3.64	6.75	4.45
ρ	0.9958	1.53e-43	1.61e-08	5.37e-06	5.23e-62	1.40e-18

no accuracy rates have mean ranks significantly different from each other. According to Friedman's test, there is a significant difference between accuracy rates. The accuracy rate of the baseline is significantly different than NIS, BPLSH, and Wilson's ENN. But there is not a significant difference between the baseline, NIS_{tuned}, DR.LSH, LSH-IS-S, and LSH-IS-F. Besides, according to both tests, there exist significant differences in terms of running time and reduction rate.

Table 4 shows the performance results of the instance selection methods on the Poker Hand data set in terms of accuracy rate, reduction rate, and running time. The Poker Hand data set has 1025010 instances and 10 features. From the results, the accuracy rates of all the instance selection methods are almost the same as the baseline. The accuracy rate difference between NIS and LSH-IS-S is 1.22%. This difference can be decreased by tuning the scaling rate parameter of NIS. NIS has the highest reduction rate of 43.09%. DR.LSH, LSH-IS-S, and LSH-IS-F deliver the lowest reduction rate by 0.22%. Besides, NIS has the lowest running time of 1.17 s. Wilson's ENN is the slowest method by 78123.37 s. Amongst methods based on hashing, LSH-IS is faster in terms of running time. As a result, the other methods get worse in terms of running time as the size of data sets increases. Consequently, NIS can yield good performance in terms of accuracy rate-reduction rate dilemma and running time on a large data set.

Table 5 shows the comparative results of the instance selection methods on some data sets in terms of accuracy rate, reduction rate, and running time. The scaling rate of NIS is 1 in 4 out of 10 data sets. The mean of the tuned scaling rate is 2.1. We have adjusted the scaling rate in one decimal digit precision. Accordingly, we point out that one does not spend a lot of time searching for a suitable scaling rate. From the results, Wilson's ENN has the highest accuracy rate in 4 out of 10 data sets. Besides, Wilson's ENN ranks first in terms of average accuracy rate as NIS ranks second. But we underline that the classification accuracy performances of the methods are almost the same. NIS yields the highest reduction rate in 6 out of 10 data sets. DR.LSH ranks second after NIS in terms of the number of the highest reduction rates. Moreover, BPLSH ranks second in terms of average reduction rate and NIS ranks first in terms of average reduction rate. Furthermore, NIS is superior to the other methods in terms of running time since it delivers the least running time on all the data sets. LSH-IS-S and LSH-IS-F are the second fastest instance selection algorithms next to NIS in terms of speed

Table 4 The performance results of the instance selection methods on the Poker Hand data set in terms of accuracy rate, reduction rate, running time

Algorithm	Performance criteria		
	Accuracy rate (%)	Reduction rate (%)	Running time (second)
1NN	59.88	–	–
NIS	58.52	**43.09**	**1.17**
BPLSH	59.32	1.03	18,594.97
DR.LSH	59.73	0.22	7335.61
LSH-IS-S	**59.76**	0.22	119.35
LSH-IS-F	59.75	0.22	29.99
Wilson's ENN	53.54	38.52	78,123.37

Best results are highlighted in boldface

Table 5 The comparative results of the instance selection methods on some data sets in terms of accuracy rate, reduction rate, and running time (the data sets are denoted by their number)

Criteria	Data set	Algorithm					
		NIS_{tuned}	BPLSH [27]	DR.LSH [26]	LSH-IS-S [25]	LSH-IS-F [25]	Wilson's ENN [14]
Accuracy rate (%)	#3	78.43	79.27	76.32	79.00	**79.64**	77.21
	#16	96.12	97.77	64.64	**97.96**	97.92	97.29
	#17	78.04	74.10	79.17	66.76	66.70	**80.13**
	#25	95.20	75.42	96.07	96.20	96.26	**97.25**
	#26	**98.09**	**98.09**	98.08	**98.09**	98.04	97.79
	#29	95.87	95.85	**95.90**	92.45	95.83	94.84
	#33	77.08	75.52	78.35	78.33	78.32	**81.01**
	#35	91.06	85.66	91.26	92.22	92.36	**92.95**
	#36	95.43	95.37	**95.45**	95.37	95.33	95.24
	#44	98.23	99.48	99.11	99.83	**99.83**	99.81
	Avg	90.36	87.65	87.44	89.62	90.02	**91.35**
Reduction rate (%)	#3	18.90	5.72	**37.92**	0.42	0.42	20.25
	#16	40.49	5.89	**98.96**	0.00	0.00	2.20
	#17	60.91	42.61	9.75	**81.78**	**81.78**	18.75
	#25	**91.53**	87.04	20.62	0.00	0.00	2.83
	#26	**19.17**	17.66	14.02	14.09	14.09	2.17
	#29	**23.62**	15.07	6.66	6.66	6.66	4.26
	#33	**54.38**	28.79	0.68	0.60	0.60	19.76
	#35	54.35	**62.12**	35.78	0.00	0.00	7.49
	#36	**19.83**	13.21	7.57	13.67	13.67	4.43
	#44	**99.01**	59.71	97.99	0.00	0.00	0.17
	Avg	**48.22**	33.78	33.00	11.72	11.72	8.23
Running time (second)	#3	**0.02**	36.71	2.72	0.43	0.30	47.21
	#16	**0.01**	21.46	0.35	0.25	0.21	29.43
	#17	**0.01**	61.74	17.74	0.25	0.22	175.34
	#25	**0.01**	10.60	5.09	0.28	0.23	38.78
	#26	**0.05**	10.39	5.40	0.51	0.56	156.68
	#29	**0.02**	9.14	1.97	0.31	0.29	66.87
	#33	**0.01**	14.39	5.74	0.29	0.25	44.97
	#35	**0.01**	7.31	2.99	0.32	0.50	19.27
	#36	**0.20**	22.26	14.29	1.27	1.23	550.92
	#44	**0.03**	7.90	1.62	1.24	1.34	426.71
	Avg	**0.04**	20.19	5.79	0.52	0.51	155.62

Best results are highlighted in boldface

comparison. Additionally, as NIS ranks first in terms of average running time, LSH-IS-S and LSH-IS-F rank second. Wilson's ENN is the slowest instance selection algorithm. Briefly, NIS rapidly processes all the data sets. According to the characteristics of data sets, as the number of instances increases, the running time of Wilson's ENN increases excessively. BPLSH and DR.LSH are affected by this situation, as well. NIS, LSH-IS-S, and LSH-IS-F are much less affected by these circumstances. BPLSH and DR.LSH are much more affected negatively by the number of dimensions compared to the number of instances. Wilson's ENN is much more impacted unfavorably compared to other algorithms by both the number of dimensions and the number of instances. Further, BPLSH is more influenced by the number of classes in comparison to the other algorithms excluding Wilson's ENN.

Figure 4 shows the accuracy and reduction rates obtained over the ten data sets according to the change in the scaling rate of NIS. The scaling rate has been picked in the range of 0.1 to 5 in increments of 0.1. According to the result, a high accuracy rate is obtained on the Nomao, HTRU2, and Human activity data sets when the scaling

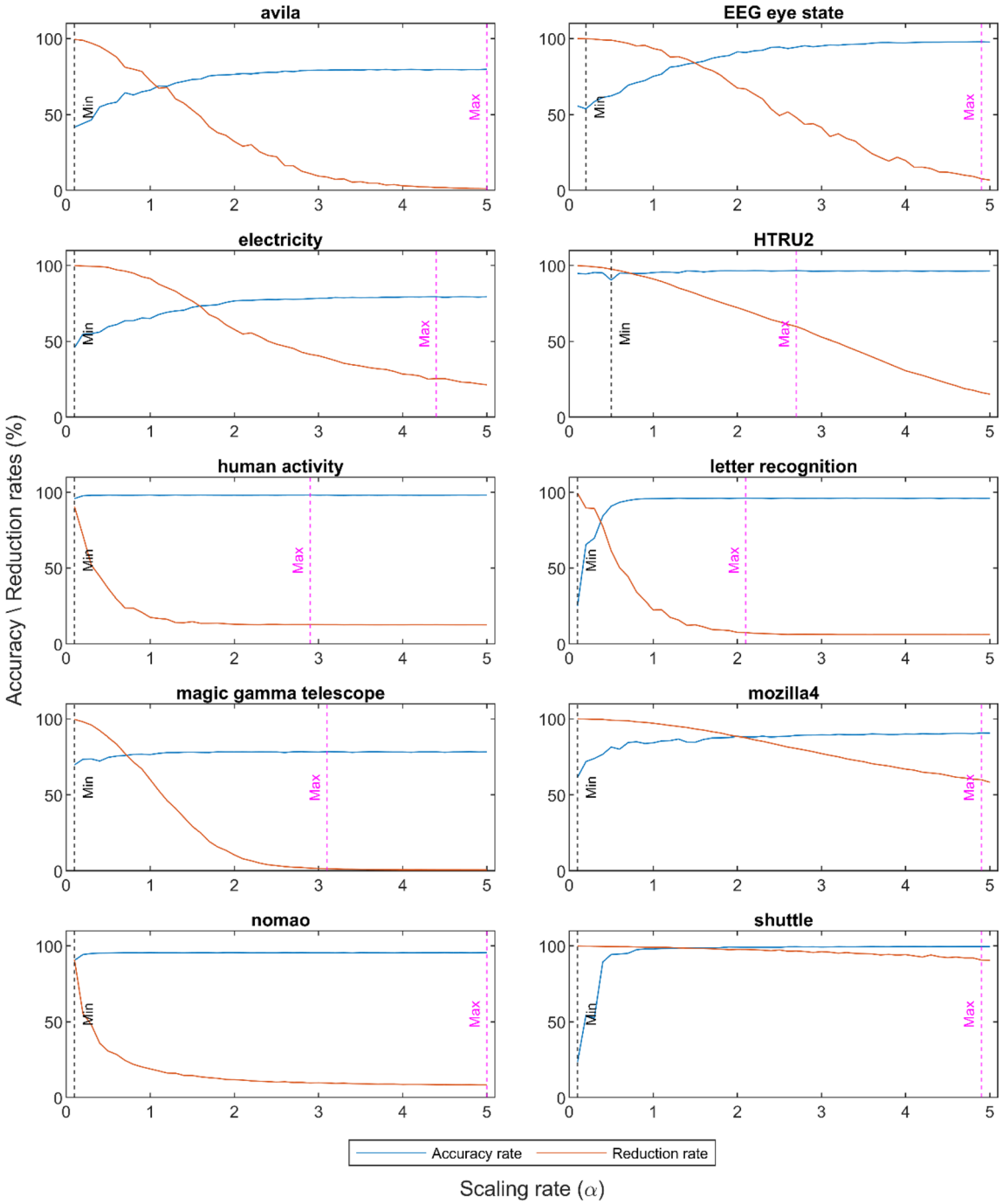

Fig. 4 The accuracy and reduction rates obtained over the ten data sets according to the change in the scaling rate of NIS

rate is 0.1. In other words, a high accuracy rate can be achieved at high reduction rates, as well. Besides, the scaling rate parameter does not affect the running time of the proposed algorithm. The number of instances or dimensions of a data set merely influences the running time of the proposed algorithm. Accordingly, both high accuracy and high reduction rates can be delivered by the fit scaling rate. Further, it is not generally needed to search for large scaling rates to get a trade-off between the accuracy rate-reduction rate. Considering the experimental results, the optimal scaling rate for data sets can be looked for in the range of 0 to 5 (zero is not included). Besides, an appropriate trade-off between accuracy rate-reduction rate can be mostly obtained when the scaling rate value is in the range of 0 to 2 in general (zero is not included). Finally, these results show that high accuracy rates cannot be also obtained at the same time on data sets when high reduction rates are reached on data sets. Therefore, either one of the

accuracy or reduction rates should be sacrificed, or a common balance point should be probed for both, as well.

In consequence, the proposed method may rapidly cope with big data in comparison with other methods. Furthermore, the proposed algorithm yields larger reduction rates on average. Besides, higher classification accuracies can be obtained with an accuracy rate—reduction rate trade-off without decreasing reduction rates too much. In addition, the idea of scaling rate is the key to overcoming large data sets. Additionally, NIS affords the best time for all data sets from different domains and characteristics. To put short it, the advantages of NIS are as follows:

- Data sets with huge volumes can be processed very fast.
- It can conduct both supervised and unsupervised learning tasks.
- It can easily manage accuracy rate—reduction rate trade-off through a single parameter.
- Simultaneously, it can provide both high accuracy and high reduction rates on the data sets.
- NIS can also deliver successful results on the imbalanced data sets.
- Better outcomes on high-dimensional data sets can be obtained by adjusting the scaling rate parameter.

The disadvantages of NIS are as follows:

- Since it selects the first one of the instances in a hyperrectangle without any criterion, the accuracy rate and reduction rate can be affected by this preference.
- Someone spends some time finding an exact scaling rate value for the data set.

6 Conclusion

In this study, we propose a new unsupervised instance selection algorithm called NIS. We have measured the success of NIS by using fifty-five data sets from different domains and compared NIS with one conventional and four state-of-the-art instance selection algorithms in recent literature. NIS yields a more desired classification accuracy-reduction rate trade-off compared to the other algorithms. The accuracy and reduction rates of NIS are 80.24% and 35.50%, respectively. The closest contestant of NIS is LSH-IS-F in terms of the accuracy-reduction trade-off. Accordingly, the accuracy and reduction rates of LSH-IS-F are 81.69% and 15.10%, respectively. Additionally, when we roughly adjust the scaling rate of NIS, we obtain the accuracy and reduction rates of NIS_{tuned} as 81.99% and 30.46%, respectively. In this case, the average scaling rate of NIS is 1.48. But the optimum scaling rate for data sets can be searched in the range of 0 to 5 (Zero is not included), according to the experimental results. NIS_{tuned} is superior to the other methods on 7 out of 55 data sets. The number of the data sets on which BPLSH, DR.LSH, LSH-IS-S, LSH-IS-F, and Wilson's ENN are superior to their own contestants is 8, 8, 8, 12, and 16, respectively. Moreover, the number of the data sets on which NIS_{tuned}, BPLSH, DR.LSH, LSH-IS-S, LSH-IS-F, and Wilson's ENN are best in their own contestants in terms of the reduction rates, is 20, 3, 3, 9, 9, and 21, respectively. The running time of NIS is quite low and can rapidly overcome huge data sets. The average, maximum, and minimum running times of NIS are 0.0332, 1.1673, and 0.0003, respectively. The closest contestant of NIS is LSH-IS-F in terms of running time. The average, maximum, and minimum running times of LSH-IS-F are 0.6663, 29.9893, and 0.0007, respectively. NIS and NIS_{tuned} are faster than the other methods on all the data sets. Although NIS is an unsupervised algorithm it can rapidly yield high accuracy rates and reduction rates over many data sets. In general, the reduction rate decreases while the accuracy rate of the methods increases. The key ability of the algorithms is that they can achieve a fit trade-off between accuracy rate, reduction rate, and speed-up. NIS ensures to be a faster algorithm and reduces more instances than other methods by allowing to achieve high accuracy rates for data sets. The future study may be the development of a supervised version of the proposed method. Thus, the accuracy rates of the algorithm on data sets can be improved more by keeping reduction rates and detecting instances from the different classes located in the same area of the space.

Data availability This study uses existing data, which is openly available at locations cited in the footnotes.

Declarations

Conflict of interests The authors have declared that no competing interests.

References

1. Saha S, Sarker PS, Al SA et al (2022) Cluster-oriented instance selection for classification problems. Inf Sci (Ny) 602:143–158. https://doi.org/10.1016/j.ins.2022.04.036
2. Olvera-López JA, Carrasco-Ochoa JA, Martínez-Trinidad JF, Kittler J (2010) A review of instance selection methods. Artif Intell Rev 34:133–143. https://doi.org/10.1007/s10462-010-9165-y
3. Garcia S, Derrac J, Cano JR, Herrera F (2012) Prototype selection for nearest neighbor classification: taxonomy and empirical study. IEEE Trans Pattern Anal Mach Intell 34:417–435. https://doi.org/10.1109/TPAMI.2011.142
4. García-Pedrajas N (2011) Evolutionary computation for training set selection. Wiley Interdiscip Rev Data Min Knowl Discov 1:512–523. https://doi.org/10.1002/widm.44

5. Hart P (1968) The condensed nearest neighbor rule (Corresp.). IEEE Trans Inf Theory 14:515–516. https://doi.org/10.1109/TIT.1968.1054155
6. Alpaydin E (1997) Voting over multiple condensed nearest neighbors. Artif Intell Rev 11:115–132. https://doi.org/10.1023/A:1006563312922
7. Gates G (1972) The reduced nearest neighbor rule (Corresp.). IEEE Trans Inf Theory 18:431–433. https://doi.org/10.1109/TIT.1972.1054809
8. Ullmann J (1974) Automatic selection of reference data for use in a nearest-neighbor method of pattern classification (Corresp.). IEEE Trans Inf Theory 20:541–543. https://doi.org/10.1109/TIT.1974.1055252
9. Ritter G, Woodruff H, Lowry S, Isenhour T (1975) An algorithm for a selective nearest neighbor decision rule (Corresp.). IEEE Trans Inf Theory 21:665–669. https://doi.org/10.1109/TIT.1975.1055464
10. Tomek I (1976) Two Modifications of CNN. IEEE Trans Syst Man Cybern SMC 6:769–772. https://doi.org/10.1109/TSMC.1976.4309452
11. Gowda K, Krishna G (1979) The condensed nearest neighbor rule using the concept of mutual nearest neighborhood (Corresp.). IEEE Trans Inf Theory 25:488–490. https://doi.org/10.1109/TIT.1979.1056066
12. Aha DW, Kibler D, Albert MK (1991) Instance-based learning algorithms. Mach Learn 6:37–66. https://doi.org/10.1007/BF00153759
13. Angiulli F (2007) Fast nearest neighbor condensation for large data sets classification. IEEE Trans Knowl Data Eng 19:1450–1464. https://doi.org/10.1109/TKDE.2007.190645
14. Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern SMC 2:408–421. https://doi.org/10.1109/TSMC.1972.4309137
15. Zhao S, Li J (2020) ELS: a fast parameter-free edition algorithm with natural neighbors-based local sets for k nearest neighbor. IEEE Access 8:123773–123782. https://doi.org/10.1109/ACCESS.2020.3005815
16. Wilson DR, Martinez TR (2000) Reduction techniques for instance-based learning algorithms. Mach Learn 38:257–286
17. Brighton H, Mellish C (2002) Advances in instance selection for instance-based learning algorithms. Data Min Knowl Discov 6:153–172. https://doi.org/10.1023/A:1014043630878
18. Li J, Zhu Q, Wu Q (2020) A parameter-free hybrid instance selection algorithm based on local sets with natural neighbors. Appl Intell 50:1527–1541. https://doi.org/10.1007/s10489-019-01598-y
19. García-Osorio C, de Haro-García A, García-Pedrajas N (2010) Democratic instance selection: a linear complexity instance selection algorithm based on classifier ensemble concepts. Artif Intell 174:410–441. https://doi.org/10.1016/j.artint.2010.01.001
20. de Haro-García A, Cerruela-García G, García-Pedrajas N (2019) Instance selection based on boosting for instance-based learners. Pattern Recognit 96:106959. https://doi.org/10.1016/j.patcog.2019.07.004
21. Cano JR, Herrera F, Lozano M (2003) Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. IEEE Trans Evol Comput 7:561–575. https://doi.org/10.1109/TEVC.2003.819265
22. de Haro-García A, Pérez-Rodríguez J, García-Pedrajas N (2018) Combining three strategies for evolutionary instance selection for instance-based learning. Swarm Evol Comput 42:160–172. https://doi.org/10.1016/j.swevo.2018.02.022
23. Dornaika F (2021) Joint feature and instance selection using manifold data criteria: application to image classification. Artif Intell Rev 54:1735–1765. https://doi.org/10.1007/s10462-020-09889-4
24. Triguero I, Peralta D, Bacardit J et al (2015) MRPR: a MapReduce solution for prototype reduction in big data classification. Neurocomputing 150:331–345. https://doi.org/10.1016/j.neucom.2014.04.078
25. Arnaiz-González Á, Díez-Pastor J-F, Rodríguez JJ, García-Osorio C (2016) Instance selection of linear complexity for big data. Knowledge-Based Syst 107:83–95. https://doi.org/10.1016/j.knosys.2016.05.056
26. Aslani M, Seipel S (2020) A fast instance selection method for support vector machines in building extraction. Appl Soft Comput 97:106716. https://doi.org/10.1016/j.asoc.2020.106716
27. Aslani M, Seipel S (2021) Efficient and decision boundary aware instance selection for support vector machines. Inf Sci (Ny) 577:579–598. https://doi.org/10.1016/j.ins.2021.07.015
28. Liu C, Wang W, Wang M et al (2017) An efficient instance selection algorithm to reconstruct training set for support vector machine. Knowledge-Based Syst 116:58–73. https://doi.org/10.1016/j.knosys.2016.10.031
29. Akinyelu AA, Ezugwu AE (2019) Nature inspired instance selection techniques for support vector machine speed optimization. IEEE Access 7:154581–154599. https://doi.org/10.1109/ACCESS.2019.2949238
30. Rico-Juan JR, Valero-Mas JJ, Calvo-Zaragoza J (2019) Extensions to rank-based prototype selection in k-nearest neighbour classification. Appl Soft Comput 85:105803. https://doi.org/10.1016/j.asoc.2019.105803
31. Ruiz IL, Gómez-Nieto MÁ (2020) Prototype selection method based on the rivality and reliability indexes for the improvement of the classification models and external predictions. J Chem Inf Model 60:3009–3021. https://doi.org/10.1021/acs.jcim.0c00176
32. Wang Z, Tsai C-F, Lin W-C (2021) Data cleaning issues in class imbalanced datasets: instance selection and missing values imputation for one-class classifiers. Data Technol Appl. https://doi.org/10.1108/DTA-01-2021-0027
33. Liu H, Motoda H (2002) On issues of instance selection. Data Min Knowl Discov 6:115–130. https://doi.org/10.1023/A:1014056429969
34. Cavalcanti GDC, Ren TI, Pereira CL (2013) ATISA: adaptive threshold-based instance selection algorithm. Expert Syst Appl 40:6894–6900. https://doi.org/10.1016/j.eswa.2013.06.053
35. Hamidzadeh J, Monsefi R, Sadoghi Yazdi H (2016) Large symmetric margin instance selection algorithm. Int J Mach Learn Cybern 7:25–45. https://doi.org/10.1007/s13042-014-0239-z
36. Hamidzadeh J, Monsefi R, Sadoghi Yazdi H (2015) IRAHC: instance reduction algorithm using hyperrectangle clustering. Pattern Recognit 48:1878–1889. https://doi.org/10.1016/j.patcog.2014.11.005
37. Leyva E, González A, Pérez R (2015) Three new instance selection methods based on local sets: a comparative study with several approaches from a bi-objective perspective. Pattern Recognit 48:1523–1537. https://doi.org/10.1016/j.patcog.2014.10.001
38. Yang L, Zhu Q, Huang J et al (2019) Constraint nearest neighbor for instance reduction. Soft Comput 23:13235–13245. https://doi.org/10.1007/s00500-019-03865-z
39. Kordos M, Blachnik M, Scherer R (2022) Fuzzy clustering decomposition of genetic algorithm-based instance selection for regression problems. Inf Sci (Ny) 587:23–40. https://doi.org/10.1016/j.ins.2021.12.016
40. Herrera-Semenets V, Hernández-León R, van den Berg J (2022) A fast instance reduction algorithm for intrusion detection scenarios. Comput Electr Eng 101:107963. https://doi.org/10.1016/j.compeleceng.2022.107963
41. Villuendas-Rey Y (2022) Hybrid data selection with preservation rough sets. Soft Comput. https://doi.org/10.1007/s00500-022-07439-4

42. Zhai J, Song D (2022) Optimal instance subset selection from big data using genetic algorithm and open source framework. J Big Data 9:87. https://doi.org/10.1186/s40537-022-00640-0