Knowl Inf Syst (2012) 30:113–133
DOI 10.1007/s10115-010-0375-z

REGULAR PAPER

Cluster-based instance selection for machine classification

Ireneusz Czarnowski

Received: 24 November 2009 / Revised: 30 June 2010 / Accepted: 13 November 2010 /
Published online: 22 January 2011

Abstract Instance selection in the supervised machine learning, often referred to as the data reduction, aims at deciding which instances from the training set should be retained for further use during the learning process. Instance selection can result in increased capabilities and generalization properties of the learning model, shorter time of the learning process, or it can help in scaling up to large data sources. The paper proposes a cluster-based instance selection approach with the learning process executed by the team of agents and discusses its four variants. The basic assumption is that instance selection is carried out after the training data have been grouped into clusters. To validate the proposed approach and to investigate the influence of the clustering method used on the quality of the classification, the computational experiment has been carried out.

Keywords Machine learning · Data mining · Instance selection · Multi-agent system

1 Introduction

Learning from examples remains the most important paradigm of the machine learning. The problem of learning from data, according to [7], can be formulated as follows: Given a dataset D, a set of hypotheses H, a performance criterion P, the learning algorithm L outputs a hypothesis $h \in H$ that optimizes P. The data D consists of N training examples, also called instances. Each example is described by a set A of n attributes. The goal of learning is to produce a hypothesis that optimizes the performance criterion. In the pattern classification application, h is a classifier (i.e. decision tree, artificial neural network, naive Bayes, k-nearest neighbor, etc.) that has been induced based on the training set D.

Research works in the field of machine learning have resulted in the development of numerous approaches and algorithms for classification problems [46,51]. One of the recent focuses of such research includes methods of selecting relevant information to be used within the

I. Czarnowski (✉)
Department of Information Systems, Gdynia Maritime University, Gdynia, Poland
e-mail: irek@am.gdynia.pl

learning process. It is obvious that removing some instances from the training set reduces time and memory complexity of the learning process [48,54]. Data reduction is especially considered as an approach to increasing effectiveness of the learning process when the available datasets are large, such as those encountered in data mining, text categorization, financial forecasting, mining of multimedia databases and meteorological, financial, industrial and science repositories, analyzing huge string data like genome sequences, Web documents and log data analysis, mining of photos and videos, or information filtering in E-commerce [23,44,54]. Finding a small set of representative instances for large datasets can result in a classification model superior to the one constructed from the whole massive data building and can help to avoid working on the whole original dataset all the time [55].

Data reduction is perceived as an important step in the knowledge discovery in databases (KDD). KDD is defined in [15] as a nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. KDD assumes that the process of data selection by data reduction plays a pivotal role in successful data mining and should not be overlooked. Much of the current research works, from the KDD domain, concentrate on scaling up data mining algorithms or scaling down the data. In the later case, the major issue is to select the relevant data and then to present it to a data mining algorithm [28]. When the mining of extremely large datasets is a difficult task, one possible solution to this problem is to reduce the number of instances [23].

The selection of the relevant data is also one of the approaches to data mining, in case the data are stored in separated and physically distributed repositories. Moving all of the data to a central site and building a single global learning model may not be feasible due to restricted communication bandwidth among the sites or high expenses involved. The selection of the relevant data in distributed locations and then moving only the local patterns can eliminate or reduce the above restrictions and speed up the distributed learning process [11].

The ultimate goal of data reduction is to reduce the size of training dataset without losing any extractable information and while simultaneously insisting that a classifier built on the reduced design data is a good or nearly as good, as a classifier built on the original dataset. The idea of data reduction has been explored and studied by many researches with the main aim to give an answer to the question why learning from the reduced data gives good results (see, for example [25,48,55]).

Data reduction can be achieved by selecting instances, features, instances and features, or discrediting features values [6]. This paper focuses on data reduction by means of instance selection, often referred to as the selection of reference vectors, instance reduction or prototype selection. The instance reduction problem is defined as a task of removing a number of instances from the original dataset D, thus producing the reduced training set S that retains essentially extractable knowledge while preserving the learning process quality. In such case, the task of learner L is to output the hypothesis $h \in H$ that optimizes performance criterion P using dataset S as a subset of the set D such that $S < D$. Ideally $S << D$.

Although a variety of instance selection methods have been so far proposed in the literature (see, for example [36,43,47,48]), no single approach can be considered as superior or guaranteeing satisfactory results in terms of the learning error reduction or increased efficiency of the learning process. While there are "zillions!" of methods for finding prototypes (see note in [5]) a method superior in certain domains is inferior in another domains [23]. Hence, searching for robust and efficient approaches to data reduction is still a lively field of research.

The contribution of the paper is twofold. First, the paper investigates a family of four novel algorithms for instance selection. The approach is based on the assumption that agent-based population learning algorithms select prototypes from clusters. Clusters are produced at the first stage of instance selection using four different approaches. Secondly, the paper shows through computational experiment how the choice of clustering procedure and the quality of the produced clusters influence the learning performance.

The paper is organized as follows. Section 2 reviews briefly instance selection algorithms. The agent-based instance selection algorithm and the description of the proposed clustering procedures are presented in Sect. 3. Section 4 contains the results of the computational experiment. Finally, in the last section, the conclusions are drawn and directions of future research are suggested.

2 Instance selection

2.1 Related work

Instance selection is one of many data reduction techniques [27]. It has been pointed out that instance selection can result in [6,9,28,48,54]:

– Increasing capabilities and generalization properties of the classification model.
– Reducing space complexity of the classification problem.
– Decreasing the required computational time.
– Diminishing the size of formulas obtained by an induction algorithm on the reduced datasets.
– Speeding up the knowledge extraction process.

Usually, instance selection algorithms are based on distance calculation between instances in the training set [47]. Methods based on other approaches, known as instance-based methods, remove an instance if it has the same output class as its k nearest neighbors, assuming that all neighborhood instances will be, later on, correctly classified [47]. Both approaches have several weaknesses. They often use distance functions that are inappropriate or inadequate for linear and nominal attributes [8,47]. Besides, there is a need to store all the available training examples in the model. To eliminate the above, several approaches have been proposed, including, for example the condensed nearest neighbor (CNN) algorithm [20], the instance-based learning algorithm 2 (IB2) [1], the instance-based learning algorithm 3 (IB3) [1], the selective nearest neighbor (SNN) algorithm [36], the edited nearest neighbor (ENN) algorithm [48], the family of decremental reduction optimization procedures (DROP1-DROP5) [48], and the instance weighting approach [54]. The other group of methods (e.g. for example: the family of four instance reduction algorithms denoted respectively IRA1-IRA4 [9], the All k-NN method [43]) try to eliminate unwanted training examples using some removal criteria that need to be fulfilled. The same principle has been mentioned in [52]. The authors of [52] conclude that if many instances of the same class are found in an area, and when the area does not include instances from the other classes, then an unknown instance can be correctly classified when only selected prototypes from such area is used.

The above reasoning also results in approaches, where the instance situated close to the center of a cluster of similar instances should be selected as a prototype (see, for example [2,47]). Such approach requires using some clustering algorithms like, for example, k-means or fuzzy k-means algorithm [14,30]. These algorithms generate cluster centers that are later considered to be the centroids, and the reduced dataset is produced. In such approach, a good reduced dataset can be obtained if the centroids are "good" representatives of clusters in the data [25]. It must be also noted that the quality of the selected centroids depends on the structure of such clusters and it is the best when they have Gaussian distribution. Furthermore, another constraint of clustering approaches is dimensionality of a dataset. When the number of features is large, the quality of the selected centroids can be poor. It means that the approach for instance selection based on the clustering must be robust independently of the structure and dimensionality of data [25,31,47,48,54]. It can be also noted that the above-discussed approaches more often are proposed as techniques for prototype extraction i.e. techniques that construct entirely new set of instances smaller, in respect to its dimensionality, than the original dataset [25].

In the IRA family of algorithms, prototypes are selected from clusters of instances and each instance has a chance to be selected as the prototype [9]. Another approach is to consider the so-called candidate instances situated close to the center of clusters and then to select the prototypes using the classification accuracy as a criterion [17].

Instance selection can also be carried out through sampling of data [49]. Sampling is a well-known statistical technique "that selects a part from a whole to make inferences about the whole" [18]. Different versions of sampling-based prototype selection, including random sampling, stratified sampling, clustering sampling, inverse sampling and others, are proposed and discussed in [26,28,32]. In [6], the so-called stratified strategy, as a variant of sampling technique, was proposed. In this case, prototypes are selected from strata using evolutionary algorithm, assuming that data are divided into disjoint strata with equal class distribution.

It was proved that the instance selection belongs to the class of NP-hard problems [19]. Thus, local search heuristics and metaheuristics like for example tabu search, simulated annealing, genetic algorithms, evolutionary algorithms, etc. seem to be the practical approach to solving the instance selection problem (see, for example [25,38,39,52]).

In [13], the selection of the relevant data is seen as the process of looking for prototypes that can be used to construct prototype-based rules.

In the literature, various hybrid data reduction methods have been also proposed. Such methods are expected to boost the effectiveness of the instance selection processes. Examples include a combination of CNN and ENN, and boosting or bootstrap approach [32]. Learning vector quantization (LVQ), a supervised learning algorithm [24], is yet another approach to improve instance selection algorithms. LVQ is used to create new prototypes rather then to select some of the existing instances as prototypes. In [21], the conjunction of the LVQ with some instance selection algorithms is discussed. The hybrid approach can be also helpful when the imbalanced datasets are observed. In such case, the class imbalance problem involved by the modifying the data distribution can help real data mining applications [45].

2.2 Taxonomies of instance selection algorithms

The instance selection methods can be classified on the basis of several different criteria. Raman and Ioerger in [35] point out that the instance selection methods can be grouped into three classes—filter methods, wrapper methods, and embedded methods. The filtering is based on random search, random sampling, or genetic algorithms. In this case, the selected instances are tested whether they come from the same distribution as the original entry dataset and whether the current reduced dataset is sufficient. The wrapper methods include boosting, where the distribution of the data is modified [35,45]. The windowing technique [34] belongs also to the wrapper group methods. Lazy learners, like the k nearest neighbors' method, belong to the embedded methods. This category of methods has the example selection strategy embedded in their learning scheme [35].

Wilson and Martinez [47] suggested that the instance selection methods can be categorized into incremental search, decremental search, and batch search. The incremental search begins with an empty set of prototypes S and adds each instance to S if it fulfills some criteria (example approaches include: CNN, the IB family, SNN). The decremental search begins with $S = D$, and successfully removes instances from S (example approaches include: ENN, the DROP family). In [18], the decremental search methods are referred to as the condensation algorithms. Finally, in the batch search mode instances fulfilling the removal criteria are removed at one go (example approaches include: kNN-based methods, the IRA family, random mutation hill climbing [38]).

The instance selection algorithms can also be classified as deterministic or non-deterministic. In deterministic ones, the final number of prototypes is controlled and determined by the user [23].

3 An approach to instance selection

3.1 Agent-based population learning algorithm for instance selection

Since instance selection belongs to the class of computationally difficult combinatorial optimization problems [12,29,41], it is proposed to apply one of metaheuristics known as the population-based algorithm [22] with optimization procedures implemented as agents within an asynchronous team of agents (A-Team). The approach is an extension of the algorithm proposed in the earlier papers of the author [9,10].

Agents working in the A-Team achieve an implicit cooperation by sharing a population of solutions to the problem at hand, called individuals. An A-Team can be defined as a set of agents and a set of memories, forming a network in which every agent remains in a closed loop [41]. All agents can work asynchronously and in parallel. Agents cooperate to construct, find and improve solutions that are read from the shared, common memory.

In our case, the shared memory is used to store a population of solutions to the instance selection problem. Each solution is represented by the set of prototypes i.e. by the compact representations of the original dataset. A feasible solution is encoded as a string consisting of numbers of selected reference vectors.

All the required steps of the proposed approach are carried out by program agents of the two types - optimizing ones and the solution manager agent. The solution manager role is to manage the population of solutions, which, at the initial phase, is generated randomly and stored in the shared memory. It is assumed that each initial solution is produced through random selection of the solitary instance from each of the earlier generated clusters.

After the initialization phase, the solution manager reads individuals from the common memory, sends them to one or more optimizing agents, and stores them back after an attempted improvement, until a stopping criterion is met. Each optimizing agent tries to improve quality of the received solutions and afterward sends them back to the solution manager, which, in turn, updates common memory by replacing a randomly selected or the worst individual (depending on the user defined evolution strategy) with the improved one. Within the proposed approach, two kinds of optimizing agents representing the tabu search and the simple local search procedures have been implemented. Pseudo-codes of both procedures are shown as Algorithms 3.1 and 3.2.

Algorithm 3.1 Local search with tabu list for instance selection.

Input: s-individual representing a solution encoded as a string consisting of numbers of selected reference instances; L-list of the instance numbers not in s; t-number of clusters of potential reference instances in s; $T = \emptyset$-tabu list; it-number of iterations an instance number stays on the tabu list.

Output: $solution$-the improved individual.

```
1. Set $i$ by drawing it at random from $\{1, 2, \ldots, t\}$.
2. Identify $j$ which is an instance number representing the $i$th cluster.
3. If $(j \subset T)$ then go to 9.
4. Set $j'$ by drawing it at random from $L$.
5. Replace an instance numbered $j$ by an instance numbered $j'$ within the $i$th cluster
   of s thus producing individual $s'$.
6. Calculate fitness of $s'$.
7. If ($s'$ is better then $s$) then ($s := s'$ AND $j$ replaces $j'$ in $L$ AND $j$ is added to $T$).
8. Remove from $T$ instances staying there for $it$ iterations.
9. If ($!terminating\_condition$) then go to 1.
10. $solution := s$.
```

Algorithm 3.2 Local search for instance selection.

Input: s-individual representing a solution encoded as a string consisting of numbers of selected reference instances; L-list of the instance numbers not in s; t-number of clusters of potential reference instances in s.
Output: $solution$-the improved individual.

```
1. Set $i$ by drawing it at random from $\{1, 2, \ldots, t\}$.
2. Identify $j$ which is an instance number representing the $i$th cluster.
3. Set $j'$ by drawing it at random from $L$.
4. Replace an instance numbered $j$ by an instance numbered $j'$ within the $i$th cluster
   of s thus producing individual $s'$.
5. Calculate fitness of $s'$.
6. If ($s'$ is better then $s$) then ($s := s'$ AND $j$ replaces $j'$ in $L$).
7. If ($!terminating\_condition$) then go to 1.
8. $solution := s$.
```

In each case, the modified solution replaces the original one if the value of its fitness has been improved. Evaluation of the solution is carried out by estimating classification accuracy of the classifier constructed, taking into account the selected instances. If, during its search, an agent has successfully improved the received solution, then it stops and the improved solution is transmitted to the solution manager. Otherwise, the agent stops searching for an improvement after having completed the prescribed number of iterations.

Application of the above agent-based population learning algorithm is preceded by grouping instances of each class into clusters. In the reported research, the following approaches to instance selection from clusters are considered:

– Selection based on the similarity coefficient.
– Selection based on the stratification strategy.
– Selection based on the modified stratification strategy,
– Selection based on the k-means clustering algorithm.

3.2 Prototype selection based on similarity coefficient

The prototype selection based on the similarity coefficient identifies clusters of instances using the approach proposed by Czarnowski and Jędrzejowicz in [9] for two-classes classification problems. In this case, at first, for each instance from the original dataset, the value of its similarity coefficient is calculated. Next, the instances with identical values of this coefficient are grouped into clusters and such procedure is carried out for instances each class independently. Finally, prototypes are selected from clusters using the population learning algorithm [22].

To show the proposed approach in a formal manner, the following notation needs to be introduced. Let x denote a training example, N denotes the number of instances in the original training set T (where $T \subset D$ and it is a set of instances available for learning), and n denotes the number of attributes. Total length of each instance (i.e. training example) is equal to $n+1$, where element numbered $n+1$ contains the class label. The class label of each example can take any value from a finite set of decision classes $C = \{c_l : l = 1, \dots, k\}$, which has cardinality k. Also, let $X = \{x_{ij} : i = 1, \dots, N; j = 1, \dots, n+1\}$ denote the matrix of $n+1$ columns and N rows containing values of all instances from T. The detailed pseudo-code of the procedure producing clusters of instances is shown below.

Algorithm 3.3 The instance grouping based on the similarity coefficient values.

Input: X—the matrix containing values of all instances from T.
Output: clusters from which prototypes can be selected.

1. Transform data instances: each $\{x_{ij}\}$ for $i = 1, \dots, N$ and $j = 1, \dots, n$ is normalized into interval [0, 1] and then rounded to the nearest integer, that is 0 or 1.
2. Calculate values:

$$s_j = \sum_{i=1}^{N} x_{ij}, \quad \text{where} \quad j = 1, \dots, n. \tag{1}$$

3. For instances from X, belonging to the class c_l (where $l = 1, \dots, k$), calculate the value of its similarity coefficient I_i:

$$\forall_{x:x_{i,n+1}=c_l} I_i = \sum_{j=1}^{n} x_{ij} s_j, \quad \text{where} \quad i = 1, \dots, N. \tag{2}$$

4. Map input vectors from X with the same value of similarity coefficient I_i into clusters.
5. Let $Y_1, \dots, Y_t$ denote the obtained clusters such that $T = \bigcup_{i=1}^{t} Y_i$ and $\forall_{i \neq j: i,j=1,\dots,t} Y_i \cap Y_j = \emptyset$.

Next, from subsets $Y_1, \dots, Y_t$ the prototypes could be selected and the reduced training set S could be produced, where initially $S = \emptyset$. The selection is based on the following rules:

– If $|Y_i| = 1$ then $S := S \cup Y_i$, where $i = 1, \dots, t$.
– If $|Y_i| > 1$ then $S := S \cup \{x_i\}$, where x_i is a reference instance selected from the cluster Y_i. the reported research this reference vector is selected by the agent-based population learning algorithm as described in Sect. 3.1.

3.3 Prototype selection based on stratification strategy

The stratification strategy has been used to prototype selection by Cano et al. in [6], where the combination of stratified strategy and evolutionary algorithm was proposed. The experiment results, presented in [6], show that the evolutionary selection of the prototypes from the strata sets can result in good generalization of the classification model. The basic idea of the stratified strategy is to divide the initial dataset into disjoint strata of equal size with equal class distribution. Using the proper number of strata can reduce significantly the size of sets from which the selection is carried out.

In this paper, the stratification is considered as a variant of clustering and from obtained sets (strata), the prototypes are selected using the agent-based population learning algorithm. The detailed pseudo-code of the procedure is shown below.

Algorithm 3.4 The instance grouping based on stratification.

```
Input: $T$—the original training set; $t$—predefined number of strata.
Output: clusters from which prototypes can be selected.

1. Map input vectors from $T$ into $t$ disjoint sets, strata with equal size and with equal
   class distribution.
2. Let $Y_1, \ldots, Y_t$ denote the obtained sets such that $T = \bigcup_{i=1}^{t} Y_i$ and $\forall_{i \neq j: i,j=1,\ldots,t} Y_i$
   $\cap Y_j = \emptyset$.
```

3.4 Prototype selection based on the modified stratification strategy

The proposed modification of the Cano et al. [6] approach assumes random division of the dataset into disjoint strata independently for each class.

Assuming that one prototype is selected from each cluster, the number of strata has a direct influence on the final number of prototypes. Since the stratification strategy is expected to produce the final prototype set with the property of having identical class distribution as the original dataset, a procedure deciding on the number of clusters for each class is proposed. The detailed pseudo-code of such procedure, denoted NCC (Number of Clusters Calculation), is shown below.

Algorithm 3.5 NCC procedure.

```
Input: $T$—the original training set; $t$—predefined number of strata.
Output: $p$—the vector that contains number of clusters for each class.

1. Map input vectors from $T$ belonging to the class $c_l (l = 1, \ldots, k)$ into disjoint
   subsets $T_l$.
2. Calculate $card(T_l)$, where $card(T_l)$ is the cardinality of $T_l$.
3. Set $M := max card(T_l)$.
4. Set $p = [p_l : l = 1, \ldots, k]$, where $p_l = \frac{card(T_l)}{M} \cdot t$.
5. Round elements of $p$ to the nearest integer.
6. Return $p$.
```

To select strata from which prototypes will be selected, it is proposed to use the agent-based population learning algorithm. The prototype selection procedure based on the modified stratification is shown below as Algorithm 3.6.

Algorithm 3.6 Strata selection procedure.

```
Input: $T$—the original training set; $t$—predefined number of strata.
Output: subset of strata from which prototypes will be selected.

1. Set $p = NCC(T, t)$, where $p$ is a vector that contains number of strata for each
   class.
```

2. Map input vectors from T belonging to the class $c_l (l = 1, \ldots, k)$ into disjoint subsets T_l.
3. Map randomly input vectors from T_l into p_l disjoint strata.
4. Let $Y_{p_l}^{(l)} (l = 1, \ldots, k)$ denote the obtained clusters such that $\forall_l$ and $\forall_{i \neq j: i,j=1,\ldots,p_l} Y_i^{(l)} \cap Y_j^{(l)} = \emptyset$, where $T = \bigcup_{l=1}^{k} \bigcup_{i=1}^{p_l} Y_i^{(l)}$ and where the total number of clusters $t^* = \sum_{l=1}^{k} p_l$.

Next, from obtained sets $Y_1^{(l)}, \ldots, Y_t^{(l)}$ the prototypes can be selected using the agent-based population learning algorithm.

3.5 Prototype selection based on the k-means clustering algorithm

The basic idea of the proposed approach is to apply the k-means clustering algorithm [29,30] to the original training dataset. Next, prototypes are selected from thus obtained clusters. The main assumption is that the clusters of data instances are produced separately for each class. It means that the number of clusters for each class must be calculated in advance. The algorithm calculating the number of clusters uses the NCC procedure.

The prototype selection based on the k-means clustering algorithm uses the agent-based population learning algorithm for the selection of prototypes. Steps 1–2 and 4 are identical as in Algorithm 3.6. Step 3 is defined as *mapping input vectors from T_l into p_l disjoint subsets (clusters) using k-means algorithm*. Finally, the prototypes are selected from the obtained clusters.

To sum up, the first presented algorithm can be considered as nondeterministic. In this case, the number of prototypes is determined by the number of clusters produced. The number of clusters depends upon the data. The next three algorithms can be classified as deterministic. Furthermore, all algorithms work in batch mode and have a feature of filter methods.

4 Computational experiment

The aim of the computational experiment was to investigate whether the clustering preceding inducement of the classifier contributes toward increasing classification accuracy. The experiment was also expected to give some insight into how the choice of clustering procedure and the quality of the produced clusters influence the classification accuracy.

4.1 Experimental data and parameter setting

To validate the proposed approach, several benchmark classification problems have been solved. Datasets for each problem have been obtained from the UCI Machine Learning Repository [3]. They include: Cleveland heart disease, credit approval, Wisconsin breast cancer, sonar problem, adult and Intelligence Customer [42]. Characteristics of these datasets are shown in Table 1.

Each benchmarking problem has been solved 30 times, and the reported values of the quality measures have been averaged over all runs. The quality measure in all cases was the correct classification ratio calculated using the 10-cross-validation approach, where at first, the available dataset was randomly partitioned into training and test sets. In the second step, each training dataset was reduced using the proposed approaches.

For the algorithm based on the modified stratification and the k-means clustering algorithm, the reduction was carried out for several different numbers of clusters. These values

Table 1 Instances used in the reported experiment

Size of the dataset	Dataset name	Number of instances	Number of attributes	Number of classes	Reported classification accuracy
small	heart	303	13	2	90.00% [12]
	sonar	208[a]	60	2	97.1% [1]
medium	credit	690	15	2	86.9% [12]
	cancer	699	9	2	97.5% [2]
large	adult	30162	14	2	84.46% [2]
	customer	24000	36	2	75.53% [38]

[a] The data set consists of 104 training and test instances

Table 2 Average number of clusters calculated by the NCC procedure for different values of the input parameter

Dataset	Number of clusters and the respective number of prototypes				
NCC parameter	10	20	30	40	50
heart	18.4 (18)	36.9 (36)	55.2 (55)	73.6 (73)	92.1 (92)
sonar	19 (19)	38 (38)	57 (57)	76 (76)	95 (95)
NCC parameter	20	40	60	80	100
credit	36.1 (36)	72.1 (72)	108.2 (108)	144.3 (144)	180.2 (180)
cancer	30.6 (30)	61.1 (61)	91.7 (91)	122.1 (122)	152.8 (152)
NCC parameter	50	100	150	200	250
customer	94.3 (94)	192.1 (192)	287.6 (283)	382.5 (382)	456.8 (456)
adult	73.3 (73)	134.3 (134)	211.4 (211)	272.1 (272)	351.4 (351)

The respective number of prototypes is shown in brackets

Table 3 Number of clusters for the stratification algorithm

Dataset	The number of clusters
heart, sonar	20, 40, 60, 80, 100
credit, cancer	35, 65, 95, 135, 165
customer, adult	80, 150, 230, 300, 400

influenced the total number of clusters calculated by the NCC procedure and hence also the final number of the selected prototypes. The values of the initial parameter of the NCC procedure and the total numbers of produced clusters with respective final number of the selected prototypes are presented, for each type of dataset, in Table 2.

In the case of the stratification algorithm, numbers of the considered strata are shown in Table 3. Recall that, in this algorithm, the number of strata (clusters) also defines the number of prototypes in the final reduced training set.

All the above input parameters have been set in a way assuring comparability with respect to the number of the selected prototypes. In the case of the clustering algorithm using the similarity coefficient, the resulting number of clusters is shown in Table 4.

All optimization agents were running for 100 iterations. The common memory size was set to 100 individuals. The number of iterations and the size of the common memory was

Table 4 Average number of clusters produced by the algorithm using the similarity coefficient

Dataset	heart	sonar	credit	cancer	customer	adult
Average number of clusters	162.5 (162)	94 (94)	184.5 (184)	133.9 (133)	172.3 (172)	431.3 (431)

The average number of the selected prototypes is shown in brackets

Table 5 Experiment results for the heart dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\|S\|/\|D\|$	Average
Full dataset	(a)	77.23	80.86	83.50	80.53	77.89	100%	80.00
SC	(b)	84.00	**85.67**	87.33	87.00	**91.21**	53%	**87.04**
SS, $t=20$	(c)	75.32	82.10	81.43	84.21	80.66	7%	80.74
SS, $t=40$	(d)	79.66	78.43	83.50	80.30	82.66	13%	80.91
SS, $t=60$	(e)	82.54	77.32	81.46	80.14	84.10	20%	81.11
SS, $t=80$	(f)	80.18	81.02	82.54	82.66	88.00	26%	82.88
SS, $t=100$	(g)	82.43	80.40	84.21	82.86	85.33	33%	83.05
MSS, $t=10$	(h)	**91.00**	83.32	**90.67**	83.33	80.91	6%	85.85
MSS, $t=20$	(i)	90.00	81.33	88.67	80.30	85.33	12%	85.13
MSS, $t=30$	(j)	89.67	82.00	89.67	86.00	87.67	18%	87.00
MSS, $t=40$	(k)	87.33	83.32	88.00	84.67	88.00	24%	86.26
MSS, $t=50$	(l)	84.67	**85.67**	86.00	82.33	91.11	30%	85.96
kCA, $t=10$	(m)	87.67	82.52	88.00	84.67	82.00	6%	84.97
kCA, $t=20$	(n)	88.33	82.67	88.00	**88.33**	85.67	12%	86.60
kCA, $t=30$	(o)	90.00	84.43	87.33	81.67	87.67	18%	86.22
kCA, $t=40$	(p)	88.33	85.00	88.33	85.00	88.00	24%	86.93
kCA, $t=50$	(r)	87.33	82.33	87.00	87.33	90.00	30%	86.80

Source: own computations

set arbitrary. The process of searching for the best solution stopped earlier than after 100 iterations in case there was no improvement of the best solution during the 3 minutes of computation. The values of these parameters have been set out experimentally at the fine-tuning phase. The starting points of k-means have been chosen randomly. The number of iterations of k-means was set arbitrary to 100. The source code was developed using the JADE framework [4] and compiled with a JAVA SE 1.6 compiler.

4.2 Computational experiment results

To evaluate the presented agent-based prototype selection algorithms, at first, the prototypes from training sets were selected. Subsequently, the reduced training set was used for inducing classifier. The following machine learning algorithms for creating classifiers have been used: C4.5 classifier with pruned leaves [34], support vector machine (WLSVM) [53] and 1NN, 10NN, Bayes Network implemented in the WEKA library [50].

Experiment results in terms of the correct classification ratio for each of the considered case are shown in Tables 5, 6, 7, 8, 9 and 10 where the following cases are covered:

– Results obtained using full dataset.

Table 6 Experiment results for the sonar dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\lvert S\rvert/\lvert D\rvert$	Average
Full dataset	(a)	94.23	75.00	73.08	**72.12**	74.04	100%	77.69
SC	(b)	94.23	75.00	75.00	40.38	**83.65**	90%	73.65
SS, $t=20$	(c)	90.32	72.43	70.54	49.32	80.43	19%	72.61
SS, $t=40$	(d)	87.64	74.67	72.23	50.46	77.43	38%	72.49
SS, $t=60$	(e)	89.65	72.32	73.08	52.34	76.32	58%	72.74
SS, $t=80$	(f)	89.06	74.08	72.41	40.15	79.04	77%	70.95
SS, $t=100$	(g)	91.20	71.07	72.48	40.15	80.30	96%	71.04
MSS, $t=10$	(h)	89.42	71.85	80.62	54.65	78.02	18%	74.91
MSS, $t=20$	(i)	92.15	73.88	79.73	58.45	77.20	37%	76.28
MSS, $t=30$	(j)	91.08	73.04	76.92	58.45	73.49	55%	74.60
MSS, $t=40$	(k)	93.12	74.88	71.15	60.06	76.37	73%	75.12
MSS, $t=50$	(l)	91.15	75.96	75.96	62.54	68.91	91%	74.91
kCA, $t=10$	(m)	80.77	70.19	80.77	58.43	79.17	18%	73.87
kCA, $t=20$	(n)	86.06	77.64	79.81	60.67	77.24	37%	76.28
kCA, $t=30$	(o)	91.35	**79.09**	80.69	59.40	75.96	55%	77.30
kCA, $t=40$	(p)	**95.67**	73.04	**80.77**	65.90	82.69	73%	**79.61**
kCA, $t=50$	(r)	94.32	72.53	77.09	61.34	83.54	91%	77.76

Source: own computations

Table 7 Experiment results for the credit dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\lvert S\rvert/\lvert D\rvert$	Average
Full dataset	(a)	82.46	86.38	75.36	85.22	84.93	100%	82.87
SC	(b)	83.33	88.70	75.22	85.94	**90.72**	27%	84.78
SS, $t=35$	(c)	80.89	83.21	79.45	78.89	80.12	5%	80.51
SS, $t=65$	(d)	83.45	84.32	77.49	77.43	80.45	9%	80.63
SS, $t=95$	(e)	83.09	83.98	76.60	80.43	82.60	14%	81.34
SS, $t=135$	(f)	84.15	84.56	72.34	81.43	82.31	20%	80.96
SS, $t=165$	(g)	83.87	86.07	77.56	80.77	81.68	24%	81.99
MSS, $t=20$	(h)	83.55	88.99	79.71	82.26	84.67	5%	83.83
MSS, $t=40$	(i)	82.57	88.84	75.94	84.23	88.47	10%	84.01
MSS, $t=60$	(j)	82.16	89.42	76.62	83.80	89.46	16%	84.29
MSS, $t=80$	(k)	84.72	**90.72**	77.39	82.51	88.89	21%	84.85
MSS, $t=100$	(l)	85.84	89.71	76.88	85.51	90.12	26%	85.61
kCA, $t=20$	(m)	86.99	87.83	84.64	**89.71**	87.12	5%	87.26
kCA, $t=40$	(n)	88.26	88.70	78.39	87.25	88.99	10%	86.32
kCA, $t=60$	(o)	86.52	88.41	**86.38**	85.65	90.14	16%	**87.42**
kCA, $t=80$	(p)	**90.72**	88.41	75.94	83.51	90.29	21%	85.77
kCA, $t=100$	(r)	87.55	88.26	85.36	82.93	89.57	26%	86.73

Source: own computations

Table 8 Experiment results for the cancer dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\|S\|/\|D\|$	Average
Full dataset	(a)	95.71	96.71	96.00	**95.57**	94.57	100%	95.71
SC	(b)	96.86	**97.43**	**96.87**	90.59	**97.44**	19%	**95.84**
SS, $t = 35$	(c)	96.32	92.43	88.32	86.76	94.54	5%	91.67
SS, $t = 65$	(d)	95.21	93.21	90.15	85.65	94.32	9%	91.71
SS, $t = 95$	(e)	96.54	94.60	92.43	89.88	96.21	14%	93.93
SS, $t = 135$	(f)	95.32	93.53	94.20	87.57	93.80	19%	92.88
SS, $t = 165$	(g)	95.87	95.93	92.62	87.65	96.03	24%	93.62
MSS, $t = 20$	(h)	98.01	95.09	93.44	90.01	94.95	4%	94.30
MSS, $t = 40$	(i)	97.86	95.72	92.15	87.29	95.72	9%	93.75
MSS, $t = 60$	(j)	**98.43**	96.29	91.01	87.01	96.44	13%	93.84
MSS, $t = 80$	(k)	**98.43**	96.01	94.15	87.15	95.44	17%	94.24
MSS, $t = 100$	(l)	98.29	96.58	93.01	87.44	96.61	22%	94.39
kCA, $t = 20$	(m)	97.32	94.83	94.54	88.30	95.71	4%	94.14
kCA, $t = 40$	(n)	97.33	95.02	95.04	89.20	94.43	9%	94.20
kCA, $t = 60$	(o)	97.77	96.97	92.15	87.54	95.09	13%	93.90
kCA, $t = 80$	(p)	97.68	97.10	93.30	90.23	96.14	17%	94.89
kCA, $t = 100$	(r)	98.33	96.51	94.54	89.65	95.57	22%	94.92

Source: own computations

Table 9 Experiment results for the customer dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\|S\|/\|D\|$	Average
Full dataset	(a)	62.92	64.30	50.96	59.83	73.32	100%	61.76
SC	(b)	67.32	70.78	58.25	**64.21**	72.43	0.72%	66.60
SS, $t = 80$	(c)	58.29	59.21	58.42	50.19	57.32	0.33%	56.69
SS, $t = 150$	(d)	58.00	59.46	57.63	52.34	60.32	0.63%	57.55
SS, $t = 230$	(e)	58.38	60.79	59.71	52.78	57.43	0.96%	57.82
SS, $t = 300$	(f)	60.08	62.17	58.92	52.50	61.54	1.25%	59.04
SS, $t = 400$	(g)	60.83	63.08	61.13	54.05	63.54	1.67%	60.53
MSS, $t = 50$	(h)	61.29	69.21	58.42	50.19	66.29	0.39%	61.08
MSS, $t = 100$	(i)	68.00	69.46	57.63	52.34	69.25	0.80%	63.33
MSS, $t = 150$	(j)	68.38	68.79	59.71	52.78	70.29	1.20%	63.99
MSS, $t = 200$	(k)	70.08	70.17	58.92	52.50	69.46	1.59%	64.23
MSS, $t = 250$	(l)	71.83	70.08	61.13	54.05	71.52	1.90%	65.72
kCA, $t = 50$	(m)	62.45	68.32	67.54	61.40	74.29	0.39%	66.80
kCA, $t = 100$	(n)	68.32	70.43	68.32	60.43	**77.25**	0.80%	68.95
kCA, $t = 150$	(o)	71.34	72.63	68.65	62.40	74.29	1.20%	69.86
kCA, $t = 200$	(p)	70.94	71.54	**69.43**	**64.21**	73.58	1.59%	69.94
kCA, $t = 250$	(r)	**72.50**	**73.05**	68.05	64.06	75.17	1.90%	**70.57**

Source: own computations

Table 10 Experiment results for the adult dataset (in %)

Reduction method	Label	1NN	10NN	Bayes network	WLSVM	C4.5	$\lvert S\rvert/\lvert D\rvert$	Average
Full dataset	(a)	79.58	80.20	73.80	62.00	82.43	100%	75.60
SC	(b)	82.43	85.00	**76.67**	69.43	85.21	1.43%	79.75
SS, $t = 80$	(c)	76.35	75.85	70.53	65.06	77.61	0.27%	73.08
SS, $t = 150$	(d)	77.65	75.79	71.15	65.44	77.40	0.50%	73.49
SS, $t = 230$	(e)	76.76	75.77	72.43	67.47	79.37	0.76%	74.36
SS, $t = 300$	(f)	77.42	75.42	70.43	65.21	80.36	0.99%	73.77
SS, $t = 400$	(g)	75.75	75.11	72.83	67.79	81.16	1.33%	74.53
MSS, $t = 50$	(h)	78.92	78.95	74.73	67.04	82.17	0.24%	76.36
MSS, $t = 100$	(i)	80.17	80.50	75.36	68.14	82.58	0.44%	77.35
MSS, $t = 150$	(j)	79.07	81.02	73.88	67.07	84.33	0.70%	77.07
MSS, $t = 200$	(k)	77.80	82.15	76.53	68.95	85.12	0.90%	78.11
MSS, $t = 250$	(l)	79.89	73.95	74.50	68.39	85.92	1.16%	76.53
kCA, $t = 50$	(m)	80.06	80.64	73.25	71.74	84.53	0.24%	78.04
kCA, $t = 100$	(n)	79.43	81.96	75.81	70.50	85.32	0.44%	78.60
kCA, $t = 150$	(o)	82.64	83.33	74.71	**72.42**	86.40	0.70%	79.90
kCA, $t = 200$	(p)	83.15	84.05	75.74	71.87	**87.43**	0.90%	80.45
kCA, $t = 250$	(r)	**85.29**	**85.29**	75.36	71.25	87.09	1.16%	**80.86**

Source: own computations

- SC—results obtained using the reduced training set produced through selection based on the similarity coefficient.
- SS—results obtained using the set of prototypes produced through selection based on the stratification strategy (these results are shown for several variants of the number of strata).
- MSS—results obtained using the set of prototypes produced through selection based on the modified stratification strategy (these results are shown for several variants of the number of strata).
- kCA—results obtained using the set of prototypes produced through selection based on the *k*-means clustering (these results are shown for several variants of the number of clusters).

Additionally, Tables 5, 6, 7, 8, 9 and 10 include the average percentage of the retained instances in the training set as well as the correct classification ratio averaged over all classifier used in each case.

From Tables 5, 6, 7, 8, 9 and 10, it is clear that the agent-based prototype selection assures better classification accuracy when compared with the case when a classifier is induced using an original, non-reduced dataset. It is also evident that the choice of the instance reduction algorithm, and specifically the underlying clustering algorithm, has an impact on the classification accuracy. The above conclusion holds true for all the considered datasets apart from the WLSVM classifier. In case of WLSVM, the data reduction results in the deterioration of the classification accuracy. This can be observed in the case of two considered classification problems. However, when the results for considered classifiers are compared, it can be concluded that the data reduction increases their performances. When the parameters are carefully selected, the performance of WLSVM is outperformed by all remaining classifiers.

Table 11 The ANOVA analysis results

Problem	Main effect A/Ho accepted	Main effect B/Ho accepted	Interaction effect/ Ho accepted
heart	–	–	–
sonar	Yes	–	Yes
credit	–	–	–
cancer	Yes	–	Yes
customer	–	–	–
adult	–	–	–

The experiment results also indicate that there is no single winning classifier although the C4.5 has been significantly better than others especially when the dataset is huge.

To confirm the above, the two-way analysis of variance (ANOVA) with the following null hypotheses has been carried out:

I: The choice of prototype selection procedure does not influence the classifier performance.
II: The choice of classifier does not influence the classification accuracy.
III: There are no interactions between both factors (i.e. the choice of the prototype selection procedure and the choice of the classifier type).

The analysis has been carried out at the significance level of 0.05. In Table 11, the ANOVA analysis results are shown. The summary results presented in Table 11 confirm that the choice of the classifier has an influence on the classification. On the other hand, the reduction in the training set through retaining prototype instances only increases, in a statistically significant manner, the classifier performance. An interaction effect between both considered factors cannot be excluded.

Taking into account the fact that the quality of the prototype selection can depend on the choice of the cluster producing procedure, it has been decided to use the non-parametric Friedman test [16] to check whether particular prototype selection procedures are equally effective independently of the kind of problem being solved.

The above test is based on weights (points) assigned to prototype selection algorithms used in the experiment. To assign weights, the 17-point scale has been used with 17 points for the best and 1 point for the worst algorithm. The test aimed at deciding among the following hypotheses:

– H_0—zero hypothesis: prototype selection algorithms are statistically equally effective regardless of the kind of the problem being solved.
– H_1—alternative hypothesis: not all working strategies are equally effective.

The analysis has been carried out at the significance level of 0.5. The respective value χ^2 statistics with 17 algorithms and 6 instances of the considered problems is 72.8, and the value of χ^2 distribution is equal to 26.3 for 16 degrees of freedom. Thus, it can be observed that not all algorithms are equally effective regardless of the kind of the problems being solved. In Figs. 1, 2, and 3, average weights for each prototype selection algorithm are shown.

From Figs. 1, 2, and 3, one can observe that the best results have been obtained by prototype selection algorithm based on the k-means clustering (see cases labeled as: ‘m’, ‘n’, ‘o’, ‘p’, ‘r’), especially in case of large datasets and high-dimensional datasets like the ‘sonar’ one. The comparable results have been obtained by prototype selection algorithm based

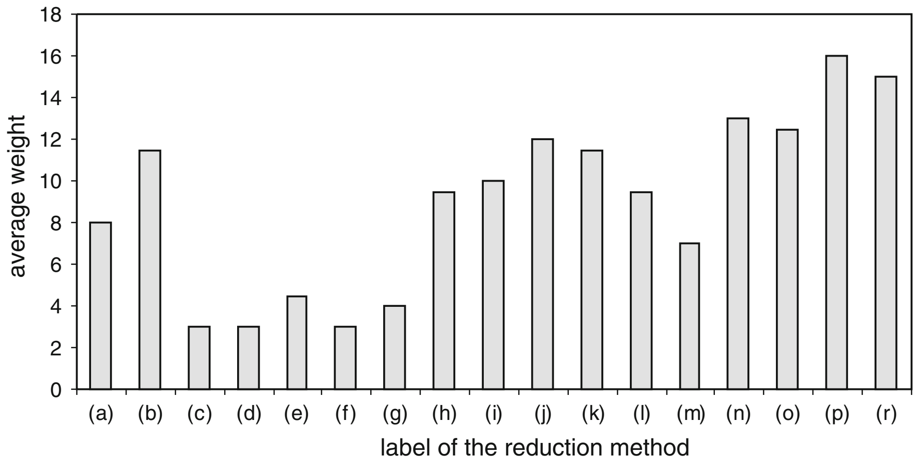

Fig. 1 The average Friedman test weights for a small-size datasets

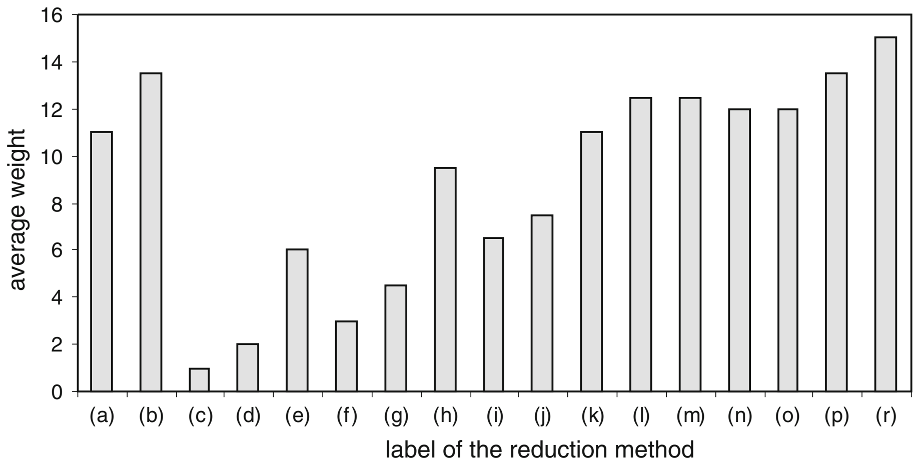

Fig. 2 The average Friedman test weights for a medium-size datasets

on the similarity coefficient (see case labeled as 'b'). This algorithm is also very effective independently of data size and dimensionality of dataset. Both algorithms outperform other approaches and this observation also holds true for the case of classifiers induced from full datasets. The worst results have been produced by the prototype selection algorithm based on stratification strategy. However, the modified stratification strategy definitely improves the results.

The proposed algorithms are also competitive in comparison with other approaches to data reduction which can be concluded from the data shown in Table 12.

The experiment results allow to observe that the number of produced clusters also influences the classifier performance. Hence, it is reasonable to assume that the number of prototypes selected in the process of data reduction can influence the classifier performance. This means that the choice of the input parameter t value is an important decision. This observation holds true for selection methods SS, MSS, and kCA.

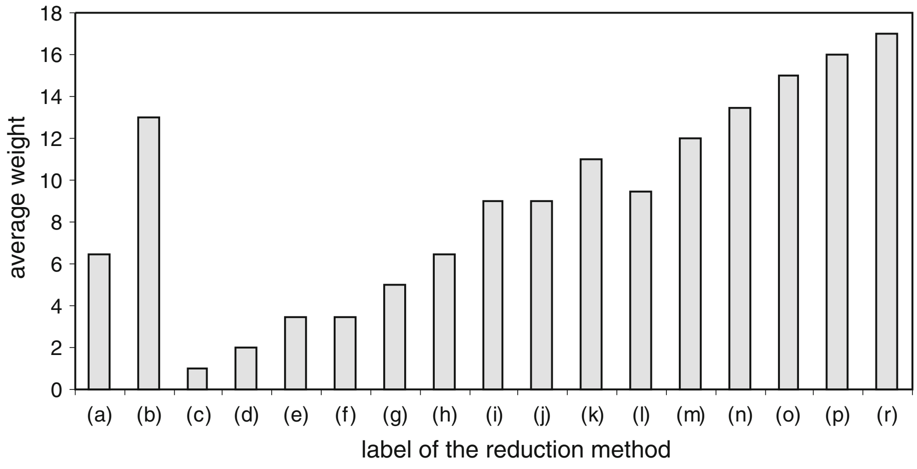

Fig. 3 The average Friedman test weights for a large-size datasets

Table 12 Comparison of different data reduction approaches (I—instance selection only; IF—instance and feature selection)

Reduction method	Type	Accuracy of classification in %					
		heart	sonar	credit	cancer	customer	adult
SC	I	**91.25**	94.23	**90.72**	97.44	72.43	85.21
kCA	I	90.00	**95.67**	**90.72**	98.33	**77.25**	**87.43**
MSS	I	91.11	93.12	**90.72**	**98.43**	71.83	85.92
SS	I	85.33	91.20	86.07	96.54	63.54	81.16
CNN [48]	I	73.95	74.12	77.68	95.71	60.43*	71.72*
SNN [48]	I	76.25	79.81	81.31	93.85	–	–
ENN [48]	I	81.11	81.79	84.49	97.00	70.65*	82.09*
3STAR [55]	I	–	73.84	86.32	–	–	–
DROP 5 [48]	I	79.84	79.88	83.91	95.71	–	–
IB3+RELIEF [35]	IF	79.94	–	71.75	73.25	–	–
RMHC [38]	IF	82.3	–	–	70.9	–	–
GA-KJ [37]	IF	74.7	55.3	–	95.5	–	–
PSO [33]	I	–	–	–	96.6	–	–
GA [33]	I	–	–	–	95.4	–	–

* Source: own computations

The above observations lead to the following two questions:

– How does the number of clusters influence the classification accuracy?
– How does the quality of the clusters influence the quality of the selected prototypes and, in consequence, the classifier performance?

Based on the computational experiment results, it can be observed that there is an interdependence between the number of selected prototypes (i.e. the reduction rate) and the classification accuracy. Its nature seems to vary depending on the dataset size. For instance,

Table 13 Silhouette width and average accuracy of classification (in %) shown with respect to comparable number of clusters)

Algorithm		heart	sonar	credit	cancer	customer	adult
SC	Silhouette width	−0.19	−0.11	−0.07	0.11	−0.13	−0.08
	Average accuracy	87.04	73.65	84.78	95.84	66.60	79.75
SS	Silhouette width	−0.42	−0.31	−0.43	−0.34	−0.28	−0.52
	Average accuracy	83.04	71.04	81.99	92.88	57.55	74.53
MSS	Silhouette width	−0.34	−0.25	−0.18	−0.32	−0.27	−0.47
	Average accuracy	85.96	74.91	85.61	94.24	63.33	76.53
kCA	Silhouette width	−0.11	−0.24	−0.08	−0.16	−0.12	−0.27
	Average accuracy	86.80	77.76	86.73	94.89	68.95	80.86

Source: own computations

for the small- and middle-size datasets, when the number of selected prototypes increases, the accuracy remains at certain level or can even fall. In case of the large datasets, the accuracy increases together with the increasing number of the selected prototypes. On the other hand, this may suggest that the relationship between the accuracy and the number of prototypes selected by a data reduction algorithm depends on the problem domain. Such conclusion was also formulated in the study dedicated to other prototype selection methods [23].

To identify the influence of the quality of clusters on the classifier performance, the silhouette technique has been used. The silhouette validation technique [40] calculates the silhouette width for each instance, average silhouette width for each cluster, and overall average silhouette width for a total dataset. Using this approach, each cluster can be represented by the silhouette width, which is based on the comparison of its tightness and separation. The average silhouette width, also called the silhouette coefficient, could be used for evaluation of the clustering quality. The silhouette value lies between -1 and 1. The higher the coefficient value the better the quality of the cluster has been obtained.

To analyze the relationships between the number of clusters, the quality of clustering and the classifier performance, the values of the silhouette coefficient versus values of the input parameter t have been observed. Based on the observation, it can be concluded that the clusters produced by applying selection method based on the k-means clustering have the highest value of the silhouette coefficient. This coincides with the fact that classifiers induced from prototypes obtained by applying selection method based on the k-means clustering (kCA) are best performers among SS, MSS, and kCA approaches.

It has also been observed that also the reduction technique based on the similarity coefficient (SC) produces very good results independently of the dataset. For the comparable instance reduction rate, SC seems better in terms of classification accuracy than other approaches. This again coincides with a high value of the silhouette coefficient characterizing clusters produced within the SC approach as shown in Table 13. The above supports the claim that the value of the similarity coefficient can be used as an important indicator in searching for a partition of instances into clusters from which prototypes are to be selected.

5 Conclusions

The paper proposes a cluster-based instance selection approach with the learning process executed by the team of agents and discusses its four variants with four different clustering procedures. It has been shown that the proposed agent-based population algorithm combined with clustering procedures is an effective instance reduction tool contributing to achieving higher quality of machine classification. The approach has been validated experimentally. Validating experiment results allow to draw the following conclusions:

– Learning from prototypes obtained within the data reduction process can produce better results than learning from full dataset.
– Agent-based approach to prototype selection extends the family of data reduction techniques adding an effective and useful alternative.
– In the data reduction based on clustering, the choice of clustering procedure is a critical factor from the point of view of the classification accuracy.
– In the data reduction based on clustering using stratification and k-means approaches, selection of the number of clusters has an influence on classification accuracy and data reduction rate. In case of the small- and middle-size datasets, it was observed that when the number of clusters increases, the accuracy remains at certain level or can even fall. In case of the large datasets, it was observed that the accuracy increases together with the increasing number of the selected prototypes.
– Instance selection based on the clustering method using the similarity coefficient approach assures better quality of learning in comparison with cases where instances are selected based on clustering using stratification or k-means approaches. It is also competitive in comparison with other data reduction approaches, so far proposed in the literature, with respect to the benchmark classification datasets.
– Clustering quality measured by the silhouette coefficient is positively correlated with the quality of classifier learning from the reduced dataset produced through data selection from clusters.

The proposed agent-based approach extends the existing range of the available techniques of data reduction. Properties of the approach should be further studied with the view of finding more effective strategies of data reduction. Future research should also be extended to implementation of the agent approach to data reduction through simultaneous instance and feature selection.

Acknowledgments This research has been supported by the Polish Ministry of Science and Higher Education with grant for years 2008-2010.

References

1. Aha DW, Kibler D, Albert MK (1999) Instance-based learning algorithms. Mach Learn 6:37–66
2. Andrews NO, Fox EA (2007) Clustering for data reduction: a divide and conquer approach. Technical Report TR-07-36, Computer Science, Virginia Tech
3. Asuncion A, Newman DJ (2007) UCI machine learning repository. University of California, School of Information and Computer Science, Irvine http://www.ics.uci.edu/~mlearn/MLRepository.html. Accessed 24 June 2009
4. Bellifemine FL, Caire G, Greenwood D (2007) Developing multi-agent systems with JADE. Wiley Series in Agent Technology, London
5. Bezdek JC, Kuncheva LI (2000) Nearest prototype classifier design: an experimental study. Int J Intell Syst 16(2):1445–1473
6. Cano JR, Herrera F, Lozano M (2004) On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining. Appl Soft Comput 6:323–332

7. Caragea D, Silvescu A, Honavar V (2003) A framework for learning from distributed data using sufficient statistics and its application to learning decision trees. Int J Hybrid Intell Syst 1(1–2):80–89
8. Chang Chin-Liang (1974) Finding prototypes for nearest neighbor classifier. IEEE Trans Comput 23(11):1179–1184
9. Czarnowski I, Jędrzejowicz P (2004) An approach to instance reduction in supervised learning. In: Coenen F (ed) Research and development in intelligent systems XX. Springer, London pp 267–282
10. Czarnowski I, Jędrzejowicz P (2009) Distributed learning algorithm based on data Reduction. In: Proceedings of ICAART 2009, Porto, Potrugal, pp 198–203
11. Czarnowski I, Jędrzejowicz P, Wierzbowska I (2009) An A-Team approach to learning classifiers from distributed data sources. Int J Intell Inf Database Syst 3(4)
12. Dash M, Liu H (1997) Feature selection for classification. Intell Data Anal 1(3):131–156
13. Duch W, Blachnik M, Wieczorek T (2005) Probabilistic distance measures for prototype-based rules. In: Proceedings of the 12th international conference on neural information processing ICONIP 2005, pp 445–450
14. Eschrich S, Ke J, Hall LO, Goldgof DB (2003) Fast accurate fuzzy clustering through data reduction. IEEE Trans Fuzzy Syst 11(2):262–270
15. Frawley WJ, Piatetsky-Shapiro G, Matheus C (1991) Knowledge discovery in databases—an overview. In: Piatetsky-Shapiro G, Matheus C Knowledge discovery in databases. AAAI/MIT Press, Cambridge
16. Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32:675–701
17. Grudziński K, Duch W (2000) SBL-PM: Simple algorithm for selection of reference instances in similarity based methods. In: Proceedings of the intelligence systems, Bystra, Poland, pp 99–107
18. Gu B, Hu F, Liu H (2001) Sampling: knowing whole from its part. In: Liu H, Motoda H (eds) Instance selection and construction for data mining. Kluwer, Dordrecht pp 21–37
19. Hamo Y, Markovitch S (2005) The COMPSET Algorithm for Subset Selection. In: Proceedings of the nineteenth international joint conference for artificial intelligence, Edinburgh, Scotland, pp 728–733
20. Hart PE (1968) The condensed nearest neighbour rule. IEEE Trans Inf Theory 14:515–516
21. Jankowski N, Grochowski M (2005) Instances selection algorithms in the conjunction with LVQ. In: Hamza MH (ed) Artificial intelligence and applications. ACTA Press, Innsbruck, p 453
22. Jędrzejowicz P (1999) Social learning algorithm as a tool for solving some difficult scheduling problems. Found Comput Decis Sci 24:51–66
23. Kim S-W, Oommen BJ (2003) A brief taxonomy and ranking of creative prototype reduction schemes. Pattern Anal Appl 6:232–244
24. Kohonen T (1986) Learning vector quantization for pattern recognition. Technical Report TKK-F-A601, Helsinki University of Technology, Espoo, Finland
25. Kuncheva LI, Bezdek JC (1998) Nearest prototype classification: clustering, genetic algorithm or random search? IEEE Trans Syst Man Cybern 28(1):160–164
26. Lazarevic A, Obradovic Z (2001) The distributed boosting algorithm. In: Proceedings ACM-SIG KDD international conference on knowledge discovery and data mining, San Francisco, pp 311–316
27. Liu H, Lu J, Yao (1998) Identifying relevant databases for multidatabase mining. In: Proceedings of Pacific-Asia conference on knowledge discovery and data mining, pp 210–221
28. Liu H, Motoda H (2001) Instance selection and construction for data mining. Kluwer, Dordrecht
29. Lu X, Xueguang S (2004) Methods of chemometrics. Science Press, Beijing
30. MacQueen JB (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley symposium on mathematical statistics and probability. University of California Press, Berkeley, vol 1, pp 281–297
31. Moise G, Sander J, Ester M (2008) Robust projected clustering. Knowl Inf Syst 14: 273–298. doi:10.1007/s10115-007-0090-6
32. Morgan J, Daugherty R, Hilchie A, Carey B (2003) Sample size and modeling accuracy of decision tree based data mining tools. Acad Inf Manage Sci J 6(2):71–99
33. Nanni L, Lumini A (2009) Particle swarm optimization for prototype reduction. Neurocomputing 72(4–6):1092–1097
34. Quinlan JR (1993) C4.5: Programs for machine learning. Morgan Kaufmann Publishers, SanMateo
35. Raman B (2003) Enhancing learning using feature and example selection. Texas A&M University, College Station
36. Ritter GL, Woodruff HB, Lowry SR, Isenhour TL (1975) An algorithm for a selective nearest decision rule. IEEE Trans Inf Theory 21:665–669
37. Rozsypal A, Kubat M (2003) Selecting representative examples and attributes by a genetic algorithm. Intell Data Anal 7(4):291–304

38. Skalak DB (1994) Prototype and feature selection by sampling and random mutation hill climbing algorithm. In: Proceedings of the international conference on machine learning, pp 293–301
39. Song HH, Lee SW (1996) LVQ combined with simulated annealing for optimal design of large-set reference models. Neural Netw 9(2):329–336
40. Struyf A, Hubert M, Rousseeuw PJ (1996) Clustering in object-oriented environment. J Stat Softw 1(4): 1–30
41. Talukdar SN, Pyo SS, Giras T (1983) Asynchronous procedures for parallel processing. IEEE Trans PAS 102(11):3652–3659
42. The European Network of Excellence on Intelligence Technologies for Smart Adaptive Systems (EUNITE)—EUNITE World Competition in domain of Intelligent Technologies (2002). http://neuron.tuke.sk/competition2 (Accessed 30 April 2002)
43. Tomek I (1976) An experiment with the edited nearest-neighbour rule. IEEE Trans Syst Man Cybern 6(6):448–452
44. Uno T (2009) Multi-sorting algorithm for finding pairs of similar short substrings from large-scale string data. Knowl Inf Syst. doi:10.1007/s10115-009-0271-6
45. Wang BX, Japkowicz N (2009) Boosting support vector machines for imbalanced data sets. Knowl Inf Syst. doi:10.1007/s10115-009-0198-y
46. Weiss SM, Kulikowski CA (1991) Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning and expert systems. Morgan Kaufmann, San Mateo
47. Wilson DR, Martinez TR (2000a) An integrated instance-based learning algorithm. Comput Intell 16:1–28
48. Wilson DR, Martinez TR (2000b) Reduction techniques for instance-based learning algorithm. Mach Learn 33(3):257–286
49. Winton D, Pete E (2001) Using instance selection to combine multiple models learned from disjoint subsets. In: Liu H, Motoda H (eds) Instance selection and construction for data mining. Kluwer, Dordrecht
50. Witten IH, Frank E (2003) Data mining: practical machine learning tools and techniques with JAVA implementations. Morgan Kaufmann, San Francisco
51. Wolper DH (2001) The supervised learning no free lunch theorems, Technical Raport. NASA Ames Research Center, Moffett Field
52. Wu Y, Ianakiew KG, Govindraju V (2001) Improvement in k-nearest neighbor classification. In: Proceedings of ICARP 2001, LNCS 2013. Springer, Berlin, pp 222–229
53. Yasser EL-Manzalawy, Honavar V (2005) WLSVM: Integrating LibSVM into Weka environment. http://www.cs.iastate.edu/~yasser/wlsvm. Accessed 20 Jan 2008
54. Yu K, Xu Xiaowei, Ester M, Kriegel H-P (2004) Feature weighting and instance selection for collaborative filtering: an information-theoretic approach. Knowl Inf Syst 5(2):201–224
55. Zhu X, Wu X (2006) Scalable representative instance selection and ranking. In: IEEE Proceedings of the 18th international conference on pattern recognition, vol 3, pp 352–355

Author Biography

Ireneusz Czarnowski holds B.Sc. and M.Sc. in Electronics and Communication Systems from Gdynia Maritime University. In 2004, he obtained Ph.D. in Computer Science from Poznań Technical University. Presently employed as assistant professor, Department of Information Systems, Faculty of Business Administration, Gdynia Maritime University. His scientific interests include combinatorial optimization, artificial intelligence, data mining, and Internet technologies.