Instance Selection: A Bayesian Decision Theory Perspective

Qingqiang Chen, Fuyuan Cao,* Ying Xing, Jiye Liang

School of Computer and Information Technology, Shanxi University, Taiyuan 030006, P.R. China
chenqq18@126.com, cfy@sxu.edu.cn, sxxying@126.com, ljy@sxu.edu.cn

Abstract

In this paper, we consider the problem of lacking theoretical foundation and low execution efficiency of the instance selection methods based on the k-nearest neighbour rule when processing large-scale data. We point out that the core idea of these methods can be explained from the perspective of Bayesian decision theory, that is, to find which instances are reducible, irreducible, and deleterious. Then, based on the percolation theory, we establish the relationship between these three types of instances and local homogeneous cluster (i.e., a set of instances with the same labels). Finally, we propose a method based on an accelerated k-means algorithm to construct local homogeneous clusters and remove the superfluous instances. The performance of our method is studied on extensive synthetic and benchmark data sets. Our proposed method can handle large-scale data more effectively than the state-of-the-art instance selection methods All code and data results are available at https://github.com/CQQXY161120/Instance-Selection.

Introduction

The massive data collected in many practical problems present new challenges for data mining and knowledge discovery approaches (Hariri, Fredericks, and Bowers 2019). For example, training the support vector machines (SVM) classifier (Chang and Lin 2011) is time-consuming on a dataset with more than one million instances (Singh, Roy, and Mohan 2016). Therefore, for these large-scale data, scalability becomes an issue. One of the most common ways to deal with this problem is instance selection (Brighton and Mellish 2002; Olvera-López et al. 2010; Hamidzadeh, Monsefi, and Yazdi 2015). It removes redundant and deleterious instances from a data set to obtain a tractable amount of instances. The significance of the instance selection method is multiple. It can reduce data storage requirements, improve the generalization performance of the classification models, and reduce the training time of the classification models.

For a classification task, the purpose of instance selection is to select a subset of instances so that a classifier trained on the subset can obtain the same or better performance than when whole instances are used. In recent years, the KNN-based instance selection, which is a kind of instance selection method that is based on the k-nearest neighbour rule, has received more attention (Garcia et al. 2012; Cavalcanti and Soares 2020; Malhat et al. 2020). The existing KNN-based instance selection methods can be classified into three categories: condensation, edition, and hybrid (Kim and Oommen 2003). The condensation methods focus on maintaining those instances located at the classification boundary and removing the instances that are far away from the boundary, among which condensed nearest neighbour (Hart 1968) is the most classical algorithm. The edition methods aim at cleaning the noisy labelled instances and outlier instances. Edited Nearest Neighbors (Wilson 1972) is a classical method in this category. The hybrid methods select the most representative instances even though the instances are at or far away from the boundary. Most of the existing methods fall into this category, such as, decremental reduction optimization procedure (DROP) (Wilson and Martinez 2000), iterative case filtering (ICF) (Brighton and Mellish 2002), Adaptive threshold-based instance selection (ATISA) (Cavalcanti, Ren, and Pereira 2013), ranking-based instance selection (RIS) (Cavalcanti and Soares 2020), and enhanced global density-based instance selection (EGDIS) (Malhat et al. 2020). Among them, RIS and EGDIS obtain excellent experimental results.

Although those instance selection methods have made significant progress, some issues need to be further study. The first issue is that these methods lack a theoretical foundation. While these methods are convenient and intuitive, there is no theoretical foundation supporting their core ideas to the best of our knowledge. The second issue is that these methods face the problem of low execution efficiency. Since these methods are derived from the k-nearest neighbour rule, they inevitably need to calculate the similarity among instances. However, calculating and storing the similarity matrix is a big overhead for large-scale instances. For example, storing a $50,000 \times 50,000$ 64-bit float matrix requires about 19G of memory, and storing a $500,000 \times 500,000$ float matrix requires about 1900G of memory.

In this paper, we propose an instance selection method called the Bayesian decision-based instance selection (BDIS) method, supported by Bayesian decision theory and percolation theory, and that can deal with instance selection problems on large-scale data. First, from the perspec-

*Corresponding author: Fuyuan Cao. Email: cfy@sxu.edu.cn.

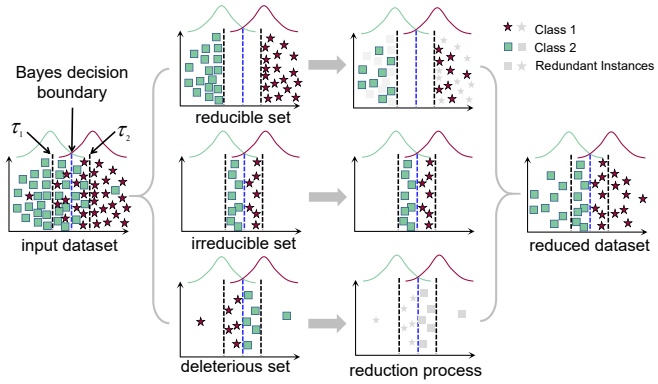


Figure 1: A framework of instance selection from the perspective of Bayesian decision theory.

tive of Bayesian decision theory, we define which instances are reducible, irreducible, and deleterious, as exemplified in Fig.1 where τ_1 and τ_2 are two parameters used to control the number of boundary instances. Coincidentally, the core ideas of the instance selection methods based on condensation, edition, and hybrid can be explained from this perspective. Then, based on the percolation theory (Meester and Roy 1996), we show that the reducible, irreducible, and deleterious instances can be found in batches by the local homogeneous cluster (LHC), which is a set of instances with the same labels. Therefore, the first step of BDIS is to find LHCs in a given data set. To achieve this goal, inspired by the approximate nearest neighbour algorithm[1], we employ an accelerated k-means (Johnson, Douze, and Jégou 2017) to divide the data set into multiple LHCs. The second step of BDIS is to select representative instances from these LHCs. Specifically, we first select those LHCs where the number of instances within LHC is between τ_1 and τ_2. Then, we select the instance closest to the centre of each LHC as the representative instance. BDIS obtains lower time and space complexity than two state-of-the-art instance selection methods. Experimental results on both synthetic and benchmark data sets verify the efficiency and effectiveness of BDIS.

The contributions of this paper include:

- We define which instances are reducible, irreducible, and deleterious from the perspective of Bayesian decision theory. The core idea of the existing KNN-based instance selection methods can be explained according to these three types of instances.
- We define local homogeneous cluster (LHC) and establish the relationship between LHC and reducible, irreducible, and deleterious instances based on percolation theory.
- We propose a method that can construct LHCs and find representative instances according to constructed LHCs with low time and space complexity.

[1]https://github.com/spotify/annoy

Instance Selection Based on Bayesian Decision Theory

Problem Setup

Suppose we are given a training data set $\mathcal{D}_{tr} := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_1}$ drawn $i.i.d$ from an unknown distribution $p(\boldsymbol{x}, y)$, where $\boldsymbol{x}_i \in X \subseteq \mathbb{R}^d$ is a training data and $y_i \in Y = \{1, 2, \cdots, m\}$ is its label. The goal of the classification problem is to learn a classifier $f(\boldsymbol{x}) : \mathbb{R}^d \rightarrow \{1, 2, \cdots, m\}$ that maps each unseen d-dimensional vector X into one of the m classes based on the training dataset. The effectiveness of the classifier is verified via a test dataset $\mathcal{D}_{te} := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_2}$, drawn $i.i.d$ from $p(\boldsymbol{x}, y)$ as well. Let $\mathcal{D}_{tr}^R$ denote a dataset after instance selection, and $f_{\mathcal{D}_{tr}}(\boldsymbol{x})$ and $f_{\mathcal{D}_{tr}^R}(\boldsymbol{x})$ represent two classifiers with the same hyperparameter build upon $\mathcal{D}_{tr}$ and $\mathcal{D}_{tr}^R$, respectively. The purpose of instance selection is to select an optimal subset $\mathcal{D}_{tr}^* \subseteq \mathcal{D}_{tr}$ to satisfy (i) $\mathcal{D}_{tr}^* = \underset{\mathcal{D}_{tr}^R \subseteq \mathcal{D}_{tr}}{\arg\max} \mid \mathbf{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}^R}(\boldsymbol{x})] - \mathbf{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}}(\boldsymbol{x})] \mid$ and (ii) $\forall \hat{\mathcal{D}}_{tr} = \underset{\mathcal{D}_{tr}^R \subseteq \mathcal{D}_{tr}}{\arg\max} \mid \mathbf{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}^R}(\boldsymbol{x})] - \mathbf{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}}(\boldsymbol{x})] \mid, |\mathcal{D}_{tr}^*| \leq |\hat{\mathcal{D}}_{tr}|$ where $|\mathcal{D}|$ represents the number of elements in the set $\mathcal{D}$.

The Proposed Methodology

Based on the above problem setup, we can find that instance selection aims to use the least training data to obtain a classifier with the highest generalization performance. As we all know, the generalization performance of any classifier for a given classification task is limited by the Bayes error rate (BER) (Fukunaga and Keinosuke 1972), which is the error rate of the Bayes optimal classifier. Let $f_{\mathcal{D}_{tr}}^B$ represent the Bayes optimal classifier over $\mathcal{D}_{tr}$, then we have $\mathrm{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}}(\boldsymbol{x})] \leq \mathrm{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}^*}(\boldsymbol{x})] \leq \mathrm{E}_{\mathcal{D}_{te}}[f_{\mathcal{D}_{tr}}^B(\boldsymbol{x})]$. Therefore, from the perspective of Bayesian decision theory, the purpose of instance selection is to make the classification model built on the selected instances set closer to the Bayesian optimal classifier by excluding some instances. So which instances should be excluded and which should be included? As far as we know, the existing studies have not systematically explained this. To fill this gap, we analyze the problem from the perspective of the Bayes posterior probability and specify which instances are reducible, irreducible, and deleterious.

As we all know, the Bayes optimal classifier achieves the minimal misclassification rate and has the form of a maximum a posterior classifier: $f^B(\boldsymbol{x}) = \underset{1 \leq i \leq m}{\arg\max} \mathrm{P}(Y = i \mid X = \boldsymbol{x})$. From the maximum posterior probability perspective, we define which instances in $\mathcal{D}_{tr}$ must be reduced, and call these instances as deleterious set,

$$\mathcal{D}_{tr}^{ds} := \{(\boldsymbol{x}, y) \in \mathcal{D}_{tr} \mid c = \arg \max_{1 \leq j \leq m} \mathrm{P}(y = j \mid \boldsymbol{x}), c \neq y\}. \tag{1}$$

$\mathcal{D}_{tr}^{ds}$ contains deleterious instances that include noisy labeled instances and instances that are misclassified by the Bayes optimal classifier, as shown in Fig.1. Existing researches indicate that these instances can interfere with

the construction of the classification model and reduce the generalization performance of the classification model (Xia et al. 2020; Zhang et al. 2020). Therefore, these deleterious instances should be removed during the instance selection process, which is the core idea of the edition based instance selection method.

Next, we define which instances are irreducible in Eqs.(2). Since the instances located at the classification boundary and possessing ground truth labels play an essential role in constructing the classifier (Heo et al. 2019), this part of the instances should not be reduced as much as possible. The methods of instance selection based on condensation are designed based on this assumption.

$$\mathcal{D}_{tr}^{is} := \{(\boldsymbol{x}, y) \in \mathcal{D}_{tr} \mid c = \arg\max_{1 \le j \le m} \mathrm{P}(y = j \mid \boldsymbol{x}), c = y, \\ 0.5 < \mathrm{P}(y = j \mid \boldsymbol{x}) \le \tau\}, \tag{2}$$

where the parameter τ is used to control the selection amount of boundary instances. The larger τ is, the more boundary instances are selected.

Remark 1. *For a given data set, deleting the deleterious instances and maintaining the irreducible instances are conducive to construct a classification model with better generalization performance.*

Finally, we define the reducible instances $\mathcal{D}_{tr}^{rs}$, that is, the instances far from the classification boundary. These instances can be suitably reduced because they belong to the corresponding class with high probability and have less impact on the construction of the classification model (Ho and Basu 2002). The formal definition of $\mathcal{D}_{tr}^{rs}$ is as follows:

$$\mathcal{D}_{tr}^{rs} := \{(\boldsymbol{x}, y) \in \mathcal{D}_{tr} \mid c = \arg\max_{1 \le j \le m} \mathrm{P}(y = j \mid \boldsymbol{x}), c = y, \\ \mathrm{P}(y = j \mid \boldsymbol{x}) \ge \tau\}. \tag{3}$$

Based on the above analysis, we transform the problem of instance selection into finding out which instances are irreducible, reducible, and deleterious for a given data set. Unfortunately, the definitions of these three types of instances are based on probabilities, making it difficult to obtain in practical problems. To overcome this problem, inspired by the percolation theory, we find a solution. Specifically, firstly we define local homogeneous cluster (LHC), a set of instances with the same labels connected by nearest neighbors, as exemplified in Fig.2. In Fig.2, a circle filled with different color represent a instance of different class, the lattice filled with same color represent a LHC. From the figure, we can observe that there are seven LHCs. For example, the lattice marked in orange is an LHC formed by 10 instances points. Then, we establish the relationship between the probability that instances in the LHC belong to the corresponding class and the size of LHC; that is, the size of an LHC is proportional to the probability that the instances in the LHC belong to the corresponding class. The formal statement is shown in Theorem 1. In this way, we can distinguish these three types of instances without knowing the probability that the instance belongs to the corresponding class.

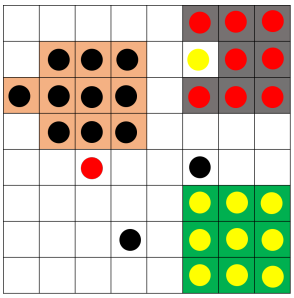


Figure 2: LHCs in a square lattice of 8×8.

Theorem 1. *For a dataset $\mathcal{D}_{tr}$, suppose $\mathcal{D}_{\mathcal{H}}$ is a LHC in the $\mathcal{D}_{tr}$ and the average maximum posterior probability of the data within $\mathcal{D}_{\mathcal{H}}$ is $p_{\mathcal{H}}$, then we have*

$$\mid \mathcal{D}_{\mathcal{H}} \mid \propto p_{\mathcal{H}}. \tag{4}$$

Before giving the proof of Theorem 1, we first provide the following lemma.

Lemma 1. *Consider a data set that generated by the probability density function $f(\boldsymbol{x})$. Let $U(\boldsymbol{x}, \delta)$ represent the δ-neighbor of a data point $\boldsymbol{x} \in \mathbb{R}^d$, σ represents the probability density of $U(\boldsymbol{x}, \delta)$, and r represents the distance of $\boldsymbol{x}$ to its nearest neighbor data point. Then when $d \ge 2$ the relationship between r and σ is*

$$r \propto \sigma^{-1}. \tag{5}$$

The skeleton of the proof of Lemma 1 is shown below.

Proof. Assume that the number of data points in the $U(\boldsymbol{x}, \delta)$ is n, and the volume of $U(\boldsymbol{x}, \delta)$ is v. Here, the point refers to a hypersphere with a certain radius. Then, the density σ of it can be given by

$$\sigma = \frac{n}{v}. \tag{6}$$

Assume that the volume of each data point is v_e, then the v is given by:

$$v = n v_e. \tag{7}$$

Combining Eqs. (6) and (7), we can obtain

$$\sigma = \frac{1}{v_e}. \tag{8}$$

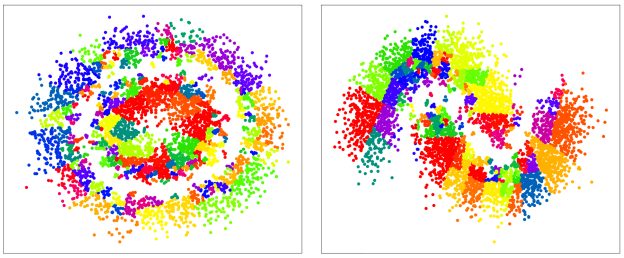
It is reasonable to assume that $r^d \propto v_e$ then

$$r \propto {v_e}^{1/d} \propto \sigma^{-1}. \tag{9}$$

□

Based on the Lemma 1, we prove the Theorem 1 as follows.

Proof. Let p denote a parameter that determines the size of the largest cluster in a system. In an infinite system the largest cluster grows with increasing parameter p, and at a critical value p_c, an infinite cluster appears. This p_c is called percolation threshold. In addition, let ξ, called the correlation length, represent the average distance of two data points



(a) Circles (b) Moons

Figure 3: LHCs formed by BDIS on two synthetic instances sets, Circles and Moons.

belonging to the same cluster. Then as $p \to p_c$, we have power laws of correlation length

$$\xi \propto |p - p_c|^{-v}, \tag{10}$$

where v is a critical exponent that depends only on general features of the topology and the local rule, and $v > 0$.

In addition, we have the power laws of average cluster size

$$S(p_c) \propto |p - p_c|^{-\gamma}, \tag{11}$$

where $S(p_c)$ represents the average cluster size over a system, γ is another critical exponent and $\gamma > 0$.

Based on these lemmas in percolation theory, we have

$$\xi \propto |p - p_c|^{-v} \Rightarrow |p - p_c| \propto \xi^{-\frac{1}{v}}, \tag{12}$$

and

$$S(p_c) \propto |p - p_c|^{-\gamma} \propto \xi^{\frac{\gamma}{v}} \propto \xi. \tag{13}$$

Furthermore, according to the definition of correlation length and nearest distance r, we have

$$\xi \propto r^{-1}. \tag{14}$$

Since $r \propto \sigma^{-1}$, we have $\xi \propto \sigma$. Therefore, the relationship between probability density σ and $S(p_c)$ is

$$S(p_c) \propto \sigma. \tag{15}$$

□

From the above theorem, we can conclude: (i) The more instances contained in an LHC, the greater the probability that the instances in the LHC belong to the corresponding class; (ii) The deleterious instances are contained in those smaller LHCs; (iii) The reducible instances are contained in the LHCs with the larger size. Therefore, we can determine which instances can be deleted by the size of LHC.

Next, we propose a method called BDIS to select representative instances for a given data set. The first step of BDIS is to use an accelerated k-means algorithm to iteratively divide the data set into multiple LHCs, as shown in Fig. 3, where a cluster of data points marked with same colors represent a LHC. For k-means implementation, we employ FAISS (Johnson, Douze, and Jégou 2017) to accelerate k-means clustering. The inspiration comes from the approximate nearest neighbor algorithm. According to the above analysis, we use two truncation thresholds (τ_1, τ_2, $0 < \tau_1 < \tau_2$) to remove the reducible and deleterious instances. Specifically, for the LHCs formed in each class of instances, we consider a LHC with less than τ_1 instances in the LHC as the LHC that must be reduced, and a LHC with more than τ_2 instances as the LHC that is reducible. In order to further compress the data set, for those LHCs with the number of instances between τ_1 and τ_2, we select the instance closest to the center of these LHC as the representative instances. For specific selection details, one can see Algorithm 1.

Algorithm 1: BDIS

Input: Training set $\mathcal{D}_{tr} = \{(\boldsymbol{x}_1, y_1) \ldots, (\boldsymbol{x}_n, y_n)\}$.
Parameters: Truncation threshold τ_1 and τ_2.
Output: Reduced set, $\mathcal{R}$.
Employ accelerated k-means algorithm to cluster $\mathcal{D}_{tr}$ into two sub-clusters;
for each sub-cluster **do**
  if the labels of data in the sub-cluster are same **then**
    We consider the sub-cluster is a LHC and record the data within it and its cluster center;
  else
    Iteratively divide the sub-cluster until it is composed of one or more LHCs.
  end if
end for
In each class of data, the LHCs with the number of instances between τ_1 and τ_2 are selected, and the instances closest to the center of these LHCs are added to $\mathcal{R}$.

Time and Space Complexity Analysis

The complexity of BDIS is reflected in the stage of obtaining LHCs. We first consider the worst case, that is, each LHC contains only one instance. The LHC construction process, in this case, is similar to constructing a binary tree with all leaf nodes containing only one instance; the time complexity is $T(n) = \mathcal{O}(n^2/\log_{10}(n))$, where n represent the number of instances for a given data set. In the best case, that is, when the BER is equal to 0, the time complexity is $T(n) = \mathcal{O}(1)$. Assuming that the inherent BER for a classification task is α, then the highest time complexity of BDIS is $T(n) = \mathcal{O}((n\alpha)^2/\log_{10}(n\alpha))$. The space complexity of the algorithm is mainly reflected in the stored data set, therefore its space complexity is $\mathcal{O}(n)$.

Experiments

To properly examine the performance of BDIS, we employ the random sampling (RS), which is one of the most classic and commonly used instance selection methods, RIS, and EGDIS as baseline methods. Specifically, for the RS method, we compare BDIS with RS10 and RS20. The RS10 and RS20 represent the RS method with 10% and 20% sampling rates, respectively. For RIS and EGDIS, we use the parameters suggested by the authors. The comparisons are carried out on multiple synthetic and 12 benchmark data sets which are available at the UCI Repository (Dua and Graff 2017). The details of the benchmark data sets are shown in

ID	Dataset	#Instances	#Attributes	#Classes
1	Optical Recognition	5,620	64	10
2	Ringnorm	7,400	20	2
3	Landsat Satellite	6,435	36	7
4	Thyroid Disease	7,200	21	3
5	Banana	5,300	2	2
6	Letter Recognition	20,000	16	26
7	MAGIC Gamma Telescope	19,020	10	2
8	Pen-Based Recognition	10,992	16	10
9	Texture	5,500	40	11
10	Twonorm	7,400	20	2
11	Shuttle	58,000	9	7
12	Skin Segmentation	245,057	4	2

Table 1: Description of data sets.

Table 1. The synthetic data sets include Moons and Circles, and their distribution is shown in Fig.4(a) and Fig.4(e). For each synthetic data sets, there are two classes of instances, which marked with different color. For each kind of synthetic data set, we generate 6,000, 60,000, 600,000, 6,000,000 and 60,000,000 instances according to their distribution, respectively. The experiments are conducted on an Intel i7-7700 CPU@3.60HZ and 48G RAM.

Evaluation Metrics

Three metrics are used to evaluate the performance of the BDIS, RS, RIS and EGDIS.

- Reduction rate (R): it measures the degree of instances reduction of the instance selection algorithm, which is defined as follows.

$$R = 1 - \frac{N_{re}}{N_{tr}}, \tag{16}$$

  where N_{tr} and N_{re} represent the number of instances in training data set and the reduced training data set, respectively.

- Classification accuracy (A): it measures the ability of a classification algorithms to classify test instances correctly using reduced training set.

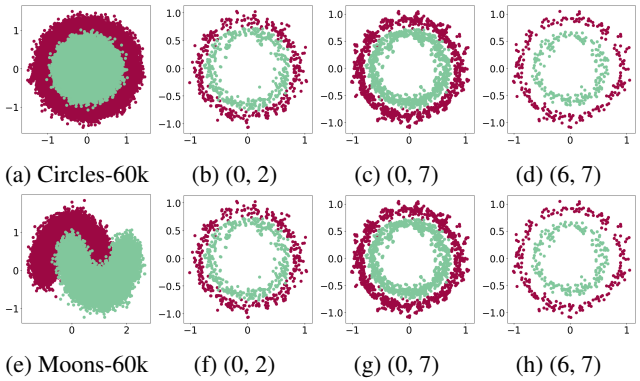
$$A = \frac{N_{cc}}{N_{te}}, \tag{17}$$

  where N_{cc} and N_{te} represent the number of correctly classified instances and test instances, respectively.

- Effectiveness (E): it measures the ability of an instance selection algorithm to balance between the reduction rate and the classification accuracy. It is computed as the product of reduction rate and classification accuracy, as given in following:

$$E = R \times A. \tag{18}$$

The classifiers employed here are three commonly used classifiers, SVM, Random forests (RF) (Breiman 2001) and k-nearest neighbor (KNN) (Cover and Hart 1967).



(a) Circles-60k (b) (0, 2) (c) (0, 7) (d) (6, 7)

(e) Moons-60k (f) (0, 2) (g) (0, 7) (h) (6, 7)

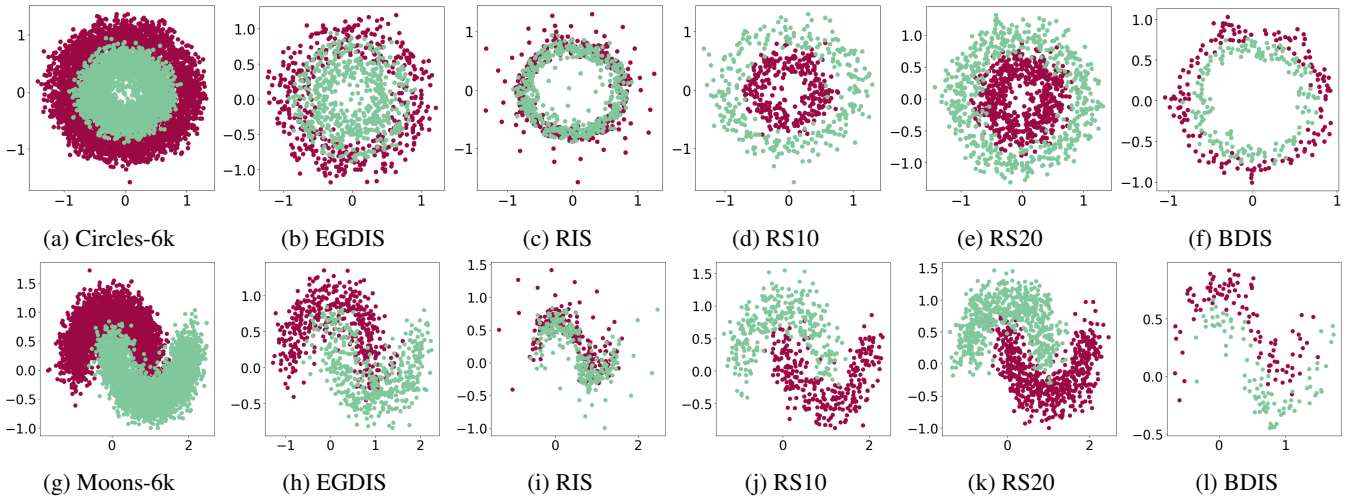
Figure 4: The influence of parameter on the BDIS. The numbers in brackets of subheadings represent the values of parameters k_1 and k_2.

Parameter Analysis

Before the comparison, we need to analyze the impact of parameters τ_1 and τ_2 on the performance of BDIS. The data sets used here are Circles and Moons with 60,000 instances, and we name these data sets Circles-60k and Moons-60k. Inspired by the (Li and Maguire 2010), we set $\tau_1 = \lfloor log_{10}(n) \rfloor + k_1$ and $\tau_2 = \lfloor log_{10}(n) \rfloor \times k_2$, where n is the number of instances for a data set, and $k_1 \in [1 - \lfloor log_{10}(n) \rfloor, n - \lfloor log_{10}(n) \rfloor]$ and $k_2 \in [\frac{1}{\lfloor log_{10}(n) \rfloor}, \frac{n}{\lfloor log_{10}(n) \rfloor}]$ are integers. We adopt the control variates method to analyze the influence of k_1 and k_2 on BDIS. Specifically, we first control k_1 unchanged and change k_2, and then control k_2 unchanged and change k_1. The instances selected by BDIS on the Circles-60k under different parameters are shown in Fig.4(b), Fig.4(c), and Fig.4(d). Comparing Fig.4(b) and Fig.4(c), we can observe that the larger the parameter k_2 is, the more non-boundary instances are selected. Comparing Fig. 4(c) and Fig 4(d), we can observe that the larger the parameter k_1 is, the greater the class boundary margin is. Similar experimental results appear on the Moons-60k, and the experimental results are shown in Fig.4(f), Fig.4(g), and Fig.4(h). From these results, we can find that when k_1 is larger, the selected instances are farther away from the boundary; when k_2 is larger, the more instances that far away from the boundary are selected. Besides, the smaller the difference between k_1 and k_2 is, the fewer instances are selected. However, too few instances may reduce the generalization performance of classifier. Therefore, in order to balance the amount of instances and the generalization performance of classifier, we empirically set $k_1 = 0$ and $k_2 = 7$ for subsequent experiments.

Experiments on Synthetic Data Sets

We empirically find that when a data set contains more than 60,000 instances, the RIS and EGDIS require more memory to run, so we compare the performance of the five methods on the Moons-6k and Circles-6k respectively. All experimental results are obtained through 10-fold cross-validation. The comparison results of different instance selection methods under A, R, and E are shown in Table 2 and Table 3



(a) Circles-6k (b) EGDIS (c) RIS (d) RS10 (e) RS20 (f) BDIS

(g) Moons-6k (h) EGDIS (i) RIS (j) RS10 (k) RS20 (l) BDIS

Figure 5: Comparison of data distribution after instance selection.

Method	Moons-6k				Circles-6k			
	A			R	A			R
	SVM	RF	KNN		SVM	RF	KNN	
EGDIS	0.970	0.943	0.920	0.836	0.940	0.917	0.871	0.828
RIS	0.960	0.945	0.925	0.904	0.939	0.900	0.897	0.856
RS10	**0.972**	**0.951**	**0.964**	0.889	0.946	0.924	0.936	0.889
RS20	0.971	0.943	0.960	0.778	0.945	**0.925**	0.936	0.778
BDIS	0.969	0.947	0.963	**0.975**	**0.948**	0.924	**0.944**	**0.957**

Table 2: Comparisons of classification accuracy (A) and reduction rate (R). The numbers in bold represent the highest A or R among the five instance selection methods.

Method	Moons-6k			Circles-6k		
	SVM	RF	KNN	SVM	RF	KNN
EGDIS	0.811	0.789	0.769	0.778	0.759	0.721
RIS	0.868	0.854	0.836	0.803	0.770	0.767
RS10	0.862	0.842	0.856	0.841	0.821	0.832
RS20	0.756	0.739	0.750	0.735	0.720	0.728
BDIS	**0.946**	**0.919**	**0.935**	**0.907**	**0.884**	**0.903**

Table 3: Effectiveness (E) comparisons. The numbers in bold represent the highest E among the five instance selection methods.

respectively. From these tables, we can observe that BDIS obtain relatively good experimental results. The instances selected by the five methods on the Moons-6k and Circles-6k are shown in Fig.5. From the figure, we can observe that the instances selected by BDIS can reduce more redundant instances.

Next, according to the distribution of Circles, we generated four datasets with 60,000, 600,000, 6,000,000, and 60,000,000 instances to test the performance of BDIS on large scale instances set. Since RIS and EGDIS require more memory, limited by computing resources, we only offer the experimental results of RS20, RS10, and BDIS. The training

Size	SVM			RF			KNN		
	RS20	RS10	BDIS	RS20	RS10	BDIS	RS20	RS10	BDIS
60,000	1.445	0.601	**0.134**	0.062	0.037	**0.019**	0.167	0.163	**0.176**
600,000	135.5	50.61	**2.796**	0.611	0.284	**0.077**	1.445	1.423	**1.325**
6,000,000	15828	5377	**54.06**	11.38	4.395	**0.772**	16.57	15.02	**13.62**
60,000,000	OT	OT	**3843**	161.4	77.28	**8.427**	202.1	171.5	**152.0**

Table 4: Comparison of training time (seconds) of classifiers on reduced Circles data sets. The OT represents that the training time of the classification model exceeds 48 hours.

Size	SVM			RF			KNN		
	RS20	RS10	BDIS	RS20	RS10	BDIS	RS20	RS10	BDIS
60,000	0.762	0.856	**0.926**	0.745	0.838	**0.899**	0.752	0.845	**0.926**
600,000	0.761	0.857	**0.933**	0.746	0.841	**0.911**	0.754	0.847	**0.932**
6,000,000	0.760	0.855	**0.935**	0.746	0.837	**0.913**	0.752	0.847	**0.935**
60,000,000	OT	OT	**0.937**	0.744	0.837	**0.917**	0.756	0.848	**0.937**

Table 5: Effectiveness comparisons on different Circle data sets.

time of classifiers on the reduced data sets and the effectiveness values of RS10, RS20, and BDIS are shown in Table 4 and 5, respectively. These results verify the effectiveness of BDIS in processing large data sets. Using similar experimental steps, we test the performance of BDIS on the Moons data sets with different numbers of instances and obtain similar experimental results.

Experiments on Benchmark Data Sets

For the top ten data sets in Table 1, we record the effectiveness values of the five instance selection methods and conduct hypothesis tests on the experimental results, as shown in Table 6. From the table, we can find that under the classification accuracy of RF and KNN, BDIS has a significant advantage over EGDIS, RIS, and RS20. Under the classification accuracy of SVM, although BDIS has no significant

Data Sets ID	SVM					RF					KNN				
	EGDIS	RIS	RS20	RS10	BDIS	EGDIS	RIS	RS20	RS10	BDIS	EGDIS	RIS	RS20	RS10	BDIS
1	0.084	0.088	0.080	**0.101**	0.091	0.643	0.482	0.631	**0.691**	**0.691**	0.809	0.831	0.774	0.857	**0.877**
2	0.347	0.304	0.399	**0.450**	0.430	0.347	0.304	0.711	**0.802**	0.758	**0.660**	0.514	0.522	0.557	0.581
3	0.189	0.182	0.197	**0.213**	0.203	0.673	0.591	0.670	0.747	**0.754**	0.683	0.634	0.695	0.764	**0.791**
4	0.760	0.710	0.742	**0.833**	0.816	0.808	0.721	0.748	**0.840**	0.823	0.679	0.679	0.745	**0.833**	0.822
5	0.688	0.639	0.722	0.809	**0.823**	0.650	0.611	0.692	0.773	**0.783**	0.637	0.616	0.707	0.790	**0.817**
6	0.081	0.033	**0.084**	0.063	0.061	0.370	0.326	0.425	0.455	**0.481**	0.659	0.678	0.696	0.713	**0.752**
7	0.257	0.380	0.520	0.585	**0.587**	0.477	0.473	0.616	**0.694**	0.689	0.464	0.422	0.624	0.696	**0.722**
8	0.085	**0.099**	0.083	**0.099**	0.097	0.706	0.760	0.693	0.773	**0.809**	0.811	0.908	0.786	0.873	**0.921**
9	0.831	0.828	0.783	0.865	**0.910**	0.674	0.566	0.646	0.704	**0.715**	0.812	0.833	0.770	0.843	**0.900**
10	0.465	0.356	0.400	0.450	**0.472**	0.820	0.655	0.743	0.822	**0.860**	0.869	0.642	0.768	0.864	**0.910**
11	0.714	OM	0.629	0.707	**0.780**	0.903	OM	0.795	0.893	**0.951**	0.903	OM	0.797	0.896	**0.953**
12	OM	OM	0.684	0.752	**0.793**	OM	OM	0.780	0.872	**0.943**	OM	OM	0.792	0.895	**0.958**
Average value (1-10)	0.379	0.362	0.401	0.447	**0.449**	0.617	0.549	0.657	0.730	**0.736**	0.708	0.676	0.709	0.779	**0.809**
Friedman test (1-10)	0.0000007					0.0000134					0.0000050				
Nemenyi test (1-10)	0.078	**0.002**	0.078	0.900	-	**0.008**	**0.001**	**0.005**	0.900	-	**0.002**	**0.001**	**0.002**	0.564	-

Table 6: Effectiveness (E) comparisons on 12 benchmark datasets. The bold font in lines 3 to 15 represent the method with the highest E among the five methods. The values in the last row represent the p value of Nemenyi test between EGDIS, RIS, RS10 and RS20 and BDIS, respectively. The bold fonts in the last row represent $p < 0.05$. The OM means that the memory required to run the corresponding algorithm exceeds 48G.

Training set	Before instance selection		After instance selection	
	A	Training time (s)	A	Training time (s)
shuttle1	0.837	525.841	0.837	0.001
shuttle2	0.837	524.441	0.837	0.002
shuttle3	0.837	532.422	0.837	0.001
shuttle4	0.837	526.449	0.837	0.002
shuttle5	0.837	523.803	0.837	0.002
shuttle6	0.837	527.755	0.837	0.001
shuttle7	0.836	525.315	0.836	0.001
shuttle8	0.837	524.552	0.836	0.001
shuttle9	0.837	531.081	0.836	0.001
shuttle10	0.837	532.175	0.837	0.002

Table 7: Comparison of training time and classification accuracy (A) of SVM classifier on different data sets before and after instance selection.

difference with EGDIS, RS10 and RS20, it obtains the highest average E value. For the 11th and 12th data sets, due to the limitation of computing resources, we only obtain partial experimental results, as shown in the Table 6. From the table, we can find that BDIS obtains the best experimental results. From these results, we can find that BDIS is superior to the compared algorithms.

To check whether BDIS can shorten the running time of the classification model without losing the generalization performance, we conduct the following experiments. We choose SVM as the classification model and shuttle as the experimental data set. For the shuttle data set, we divide it into 10 disjoint subsets, and select one of them as test set and the remaining nine as the training set in turn. In this way, we obtain 10 training sets, shuttle1, shuttle2,..., shuttle10. For each training set, we train SVM on the training set before and after instance selection, respectively, and record the training time and classification accuracy on the corresponding test set. The experimental results are shown in Table 7. From the table, we can find that the classification accuracy does not change significantly for each training set, but the training time of SVM on the reduced data set is significantly shortened. These experimental results further verify the effectiveness of the BDIS algorithm.

Conclusions

In this paper, we defined which instances are reducible, irreducible, and deleterious for a given data set from the perspective of Bayesian decision theory. Then, based on the percolation theory, we established the relationship between the three kinds of instances and LHC. Finally, we proposed a method that can construct LHC and find representative instances from the data set faster and less memory-consuming. Besides, we found that the core idea of the instance selection methods based on the k-nearest neighbor rule is to distinguish the reducible, irreducible, and deleterious instances. In the experimental analysis, we compared BDIS with classical and state-of-the-art instance selection methods on synthetic and benchmark sets. The experimental results verified the effectiveness and efficiency of BDIS.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (2020AAA0106100) and the National Natural Science Foundation of China (61976128).

References

Breiman, L. 2001. Random forests. *Machine learning*, 45(1): 5–32.

Brighton, H.; and Mellish, C. 2002. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery*, 6(2): 153–172.

Cavalcanti, G. D.; Ren, T. I.; and Pereira, C. L. 2013. ATISA: Adaptive threshold-based instance selection algorithm. *Expert Systems with Applications*, 40(17): 6894–6900.

Cavalcanti, G. D.; and Soares, R. J. 2020. Ranking-based instance selection for pattern classification. *Expert Systems with Applications*, 150: 113269.

Chang, C.-C.; and Lin, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3): 1–27.

Cover, T.; and Hart, P. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1): 21–27.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences. Accessed 2021-05-15.

Fukunaga; and Keinosuke. 1972. *Introduction to statistical pattern recognition*. Academic Press.

Garcia, S.; Derrac, J.; Cano, J.; and Herrera, F. 2012. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3): 417–435.

Hamidzadeh, J.; Monsefi, R.; and Yazdi, H. S. 2015. IRAHC: instance reduction algorithm using hyperrectangle clustering. *Pattern Recognition*, 48(5): 1878–1889.

Hariri, R. H.; Fredericks, E. M.; and Bowers, K. M. 2019. Uncertainty in big data analytics: survey, opportunities, and challenges. *Journal of Big Data*, 6(1): 1–16.

Hart, P. 1968. The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory*, 14(3): 515–516.

Heo, B.; Lee, M.; Yun, S.; and Choi, J. Y. 2019. Knowledge distillation with adversarial samples supporting decision boundary. In *AAAI*, 3771–3778.

Ho, T. K.; and Basu, M. 2002. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3): 289–300.

Johnson, J.; Douze, M.; and Jégou, H. 2017. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734*.

Kim, S.-W.; and Oommen, B. J. 2003. A brief taxonomy and ranking of creative prototype reduction schemes. *Pattern Analysis & Applications*, 6(3): 232–244.

Li, Y.; and Maguire, L. 2010. Selecting critical patterns based on local geometrical and statistical information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6): 1189–1201.

Malhat, M.; El Menshawy, M.; Mousa, H.; and El Sisi, A. 2020. A new approach for instance selection: Algorithms, evaluation, and comparisons. *Expert Systems with Applications*, 149: 113297.

Meester, R.; and Roy, R. 1996. *Continuum percolation*, volume 119. Cambridge University Press.

Olvera-López, J. A.; Carrasco-Ochoa, J. A.; Martínez-Trinidad, J. F.; and Kittler, J. 2010. A review of instance selection methods. *Artificial Intelligence Review*, 34(2): 133–143.

Singh, D.; Roy, D.; and Mohan, C. K. 2016. DiP-SVM: distribution preserving kernel support vector machine for big data. *IEEE Transactions on Big Data*, 3(1): 79–90.

Wilson, D. L. 1972. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems Man Cybernetics-Systems*, SMC-2(3): 408–421.

Wilson, D. R.; and Martinez, T. R. 2000. Reduction techniques for instance-based learning algorithms. *Machine learning*, 38(3): 257–286.

Xia, X.; Liu, T.; Han, B.; Wang, N.; Gong, M.; Liu, H.; Niu, G.; Tao, D.; and Sugiyama, M. 2020. Part-dependent Label Noise: Towards Instance-dependent Label Noise. In *NIPS*.

Zhang, Z.; Zhang, H.; Arik, S. O.; Lee, H.; and Pfister, T. 2020. Distilling effective supervision from severe label noise. In *CVPR*, 9294–9303.