Cost-Constrained Data Acquisition for Intelligent Data Preparation

Xingquan Zhu, *Member*, *IEEE*, and Xindong Wu, *Senior Member*, *IEEE*

Abstract—Real-world data is noisy and can often suffer from corruptions or incomplete values that may impact the models created from the data. To build accurate predictive models, data acquisition is usually adopted to prepare the data and complete missing values. However, due to the significant cost of doing so and the inherent correlations in the data set, acquiring correct information for all instances is prohibitive and unnecessary. An interesting and important problem that arises here is to select what kinds of instances to complete so the model built from the processed data can receive the "maximum" performance improvement. This problem is complicated by the reality that the costs associated with the attributes are different, and fixing the missing values of some attributes is inherently more expensive than others. Therefore, the problem becomes that given a fixed budget, what kinds of instances should be selected for preparation, so that the learner built from the processed data set can maximize its performance? In this paper, we propose a solution for this problem, and the essential idea is to combine attribute costs and the relevance of each attribute to the target concept, so that the data acquisition can pay more attention to those attributes that are cheap in price but informative for classification. To this end, we will first introduce a unique Economical Factor (EF) that seamlessly integrates the cost and the importance (in terms of classification) of each attribute. Then, we will propose a cost-constrained data acquisition model, where active learning, missing value prediction, and impact-sensitive instance ranking are combined for effective data acquisition. Experimental results and comparative studies from real-world data sets demonstrate the effectiveness of our method.

Index Terms—Data mining, intelligent data preparation, data acquisition, cost-sensitive, machine learning, instance ranking.

1 Introduction

Data mining techniques have been widely employed in various applications. A general mining procedure usually involves the following four major steps:

1. collecting data,
2. transforming/cleansing the data,
3. model building, and
4. model deployment/monitoring [1], [2].

Most practitioners have agreed that data preparation, in steps 1 and 2 above, consumes the majority of the data mining cycle time and budget [3]. As a result, intelligent data preparation has attracted significant attention, in which data integrity and quality control [4] issues play more and more important roles for effective mining. A common real-world problem is that many data mining applications are characterized by the collection of incomplete data in which some of the attribute values are missing (due to reasons such as privacy, data ownership, or unavailability) but can be acquired at a cost. For example, a patient's record in hospital A may have the diagnoses of one particular disease missing, but this information is likely available in the patient's record in hospital B and, therefore, can be acquired at a cost. Similarly, the credit card companies have data on customer transactions with their cards, but usually do not have data on the same customer's transactions with other cards. An important issue here for intelligent data preparation is how to acquire incomplete data items for better data quality, integrity and, consequently, better results.

To induce a decision tree or build other classification models, in the context of incompletely-specified training examples, simply ignoring instances with missing values leads to inferior model performance [5], and the model specifically designed for classifying incomplete data [6], [7], [8], [9] also suffers from a decrease of accuracy in describing the concept. Accordingly, many research efforts [10], [11] have been conducted to prepare the data through predicting missing attribute values (which is called imputation in statistics) by using other instances in the data set, such as filling in missing attribute values with the attributes' most common values [12], using a Bayesian formalism [13] or a decision-tree [10] to predict the missing values, or adopting clustering and regression techniques [14]. Quinlan [15] uses the entropy and splits the examples with missing attribute values to all concepts while constructing a decision tree. These methods are efficient in their own scenarios, but their reliability is still the biggest concern because the accuracy of predicting the missing values could be very low in many situations [16] (and it is not surprising that many attribute values simply cannot be predicted at all). Accordingly, users are reluctant to adopt these "automatic" imputation methods if they are very serious with their data, e.g., hospital or Census Bureau data. On the other hand, widely existed missing values actually force users to complete some of the missing items for many data mining purposes. For real-world applications, acquiring information for all incomplete instances is completely out of the question and impractical given the amount of human labor and costs involved. These contradictions raise a new research issue: how many, and which incomplete instances need additional

- *X. Zhu is with the Department of Computer Science, University of Vermont, 33 Colchester Ave., Votey 377, Burlington, VT 05401. E-mail: zqzhu@cs.uvm.edu.*
- *X. Wu is with the Department of Computer Science, University of Vermont, 33 Colchester Ave., Votey 351, Burlington, VT 05401. E-mail: xwu@cs.uvm.edu.*

Manuscript received 21 Nov. 2004; revised 29 Dec. 2004; accepted 17 Apr. 2005; published online 19 Sept. 2005
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDESI-0472-1104.

data, so that the learned model can maximize its performance? In reality, this problem is complicated by the fact that the costs associated with the attributes are different, and fixing missing values of some attributes is much more expensive than others. Therefore, the problem becomes that given a fixed budget, what kinds of instances should be selected for processing, so that the learner built from the processed data set can maximize its performance. We call this problem cost-constrained data acquisition because a good solution should consider the attribute costs and acquire information with a budget constraint.

2 Related Work

Given a target problem, selecting "good" instances to facilitate the problem solving has been a major issue for many research areas, which include:

1. instance selection for effective learning,
2. instance selection for large data sets, and
3. instance selection for active learning.

To select "good" examples for effective learning, Wilson [17] used a *k*-nearest neighbor (*k-NN*) classifier to select instances that were then used to form a *1-NN* classifier, and only those instances that were classified correctly by the *k-NN* were retained for the *1-NN*. Based on this scheme, many instance selection schemes have been developed for effective learning [18]. A comprehensive overview of the research on this topic can be found in [19]. Due to the fact that the quality of the training data vitally determines the accuracy of the classifier, the idea of selecting "good" examples has also been applied to other types of classifiers, where various kinds of selection approaches, such as near misses and the most-on-point cases, are adopted to find the examples that can improve the effectiveness of the classifiers [20].

When the size of the training set becomes considerable, it will inevitably raise the challenge for induction algorithms to scale up for large data sets. Many existing efforts alleviate this problem by selecting some "good" instances to feed the learner. The underlying assumption is that having access to massive amounts of data does not necessarily mean that induction algorithms must use them all, and by carefully selecting some good instances from massive data, we shall improve (at least do not decrease) the learner's ability, such as improving the classification accuracy or enhancing the performance of the classifiers in dealing with large data sets [21]. As illustrated by Lewis and Catlett [22], sampling instances using an estimate of classification certainty drastically reduces the amount of data needed to learn a concept.

Active learning [23], [24], [25], [26] (also called pool-based sample selection), on the other hand, provides goal-directed data selection, in which the learner has the freedom to select which data points to be added to its training set. The intuitive assumption is that labeling training examples is often expensive, an active learner may begin with a very small number of labeled examples, carefully select a few additional examples for which it requests labels, learn from the results of that request, and then using its newly-gained knowledge, carefully choose which examples to request next. In this way, the active learner aims to reach high performance using as few labeled examples as possible (in comparison with randomly selecting instances to label). The most popular solutions to handle this problem are to label the instances that likely confuse the learner [26], or receive the most disagreement from the committee [24]. Preparing information for these instances likely provides more fresh knowledge to the learner and, therefore, may possibly result in the lowest error on future test examples.

Unfortunately, all methods above are based on the data with complete information, which are essentially different from our objective. In [27], an active data acquisition approach has been proposed to complete information for instances with missing values. This method is the most close to our paper. However, the disadvantage of this approach is twofold: 1) it did not consider the correlation between the attributes and the class and 2) the attribute, which receives the most disagreements from different imputation methods, is regarded as the most important attribute for completion. Clearly, if one attribute is totally independent of others, the predictions from all imputation methods on this attribute are likely random (with the maximum disagreements), but providing information for this attribute has a very limited contribution to the system. In [28], an impact-sensitive instance ranking mechanism has been proposed to rank suspicious instances with erroneous attribute values. Although this method was not originally designed for data sets with missing values, it actually motivates us that a data acquisition approach should take the attribute importance into consideration and put a preference on informative attributes (the attributes with a high relevance to the target concept). Motivated by this observation, we have proposed a data acquisition approach with impact-sensitive instance ranking [29]. It takes the attribute importance into consideration, and selects the incomplete instances with the most missing values on the important attributes for preparation.

Existing data acquisition efforts assume that acquiring correct information for each incomplete instance has the same expense, no matter how many attributes of the instance actually contain missing values. This is rarely the case in reality, where finding information for some attributes costs much more than for others (which is a well-known fact in cost-sensitive classification [30], [31]) and, therefore, the costs to acquire complete information for different instances are inherently different, depending on how many and which attribute values are missing. Take the Heart disease data set from the UCI data repository [32] as an example, where monitoring the patient's maximum heart rate (attribute 8), which costs \$102.90, is 13 times more expensive than checking the patient's serum cholesterol (attribute 5), which costs \$7.27.

In the context that the attribute cost is a big concern, a data acquisition algorithm cannot just put priority on informative attributes because these attributes might be too expensive compared with their contributions. We will have to develop a new measure to compromise the attribute importance and its cost for better results. Research efforts in cost-sensitive classification have already addressed related problems [30], [31], [33], [34], where the test cost is used to develop systems which cost less in the test phase but still maintain good accuracy. In a naïve sense, just imagine that you are a doctor with a decision theory of your own domain of expertise. To determine the disease of a new patient, you will have to evaluate his/her symptoms (attribute values) with a certain amount of cost for each symptom. The "optimum" decision is the one which costs the least to diagnose the patient but still reach the correct decision. To this end, many heuristic mechanisms, e.g., ICET [31], EG2 [33], CS-ID3 [34], have been developed, where the essential

idea is to use a greedy strategy to modify a normal decision theory by taking attribute costs into consideration. Although these approaches are not originally designed to solve our problem here, they actually inspire us on how to integrate the attribute cost and importance for better results.

Intuitively, we believe that three important issues should be taken into consideration when designing cost-constrained data acquisition algorithms:

1. Missing values of different attributes are inherently different, in terms of costs and their relevance to the target concept. We should pay more attention to the economical attributes, which are important for classification but cost relatively less.
2. Some incomplete instances may already contain enough information for model construction, and putting efforts to them is unlikely to receive much improvement.
3. Some missing values in an instance might be predictable by other instances, so we may put less effort to them when acquiring data for completion.

In this paper, we report our recent research efforts in resolving the above concerns. We propose a unique *Economical Factor* (*EF*) to comprise the attribute importance and cost, and explore the economical attributes for data acquisition purposes in Section 3. In Section 4, we discuss a new cost-constrained data acquisition mechanism that combines active learning, attribute prediction, and the *EF* measure. We perform comparative studies in Section 5. Concluding remarks are given in Section 6.

Throughout the paper, we use the notion that each instance I_k is represented by a pair of ordered vectors $(A_1, \ldots, A_i, \ldots, A_M, c)$ and $(C_{A_1}, \ldots, C_{A_M})$, where c is the target concept (class label), A_i and C_{A_i} denote the ith attribute and its cost, respectively, and M is the number of attributes. For each attribute, say A_i, it has M_i values, denoted by $a_{i,1}, \ldots, a_{i,M_i}$. $p(a_{i,j})$ means the probability that attribute A_i has value $a_{i,j}$, $p(c_l)$ is the probability of the class c_l, and $p(c_l|a_{i,j})$ is the probability of class c_l conditioned by attribute A_i with value $a_{i,j}$.

3 Attribute Evaluation

3.1 The Attribute Discrimination Measure

The problem of measuring the attribute *Discrimination Efficiency* (*DE*) has received much attention in the literature, and the essential goal is to evaluate the relevance between the attribute and users' tasks at hand (in the context of classification, it is the relevance between the attribute and the target concept). There are many measures for estimating attributes' discrimination efficiency, and several most popular measures for classification problems include the Gini index [9] and Information-gain Ratio (*IR*) [15]. Basically, most of the existing *DE* measures are impurity-based, meaning that they measure impurity of the class value distribution. They assume the conditional (upon the class) independence of the attributes, and evaluate each attribute separately by measuring impurity of the splits resulting from partitions of the learning instances according to the values of the evaluated attribute (without taking the context of other attributes into account). The general form of all impurity-based measures is defined by (1):

$$DE(A_i) = i(c) - \sum_{j=1}^{M_i} p(a_{i,j}) i(c|a_{i,j}), \tag{1}$$

where $i(c)$ is the impurity of the class values before the split, and $i(c|a_{i,j})$ is the impurity of the class values after the split on attribute $A_i = a_{i,j}$. By subtracting weighted impurity of the splits from the impurity of unpartitioned instances, we measure the gain in the purity of class values resulting from each split. Larger values of $DE(A_i)$ imply better splits and, therefore, better attributes (in terms of the target concept).

Although we have several choices in selecting different *DE* measures, we adopt the information-gain ratio *IR* [15], defined by (2), in our system. The reason is that *IR* is implemented in C4.5 and is the most often used impurity-based measure. If *IR* successfully solves our problems, other impurity-based measures could be integrated in the same way. Equation (2) gives the definition of the *IR* for attribute A_i, denoted by IR_i:

$$IR_{A_i} = \frac{\sum_{l=1}^{|c|} p(c_l) \log p(c_l) - \sum_{j=1}^{M_i} \sum_{l=1}^{|c|} p(c_l|a_{i,j} \log p(c_l|a_{i,j})}{\sum_{j=1}^{M_i} p(a_{i,j}) \log p(a_{i,j})}. \tag{2}$$

With (2), the information gain part (the numerator) tries to maximize the difference of entropy (which serves as the impurity function) before and after the split. To prevent an excessive bias caused by the number of attribute values, the information gain was normalized by the attribute's entropy. The higher the IR_{A_i}, the more important the attribute A_i is.

3.2 Attribute Economical Factor

With the *DE* measure above, we can explicitly and quantitatively evaluate the importance of each attribute. The higher the *DE* value, the more informative the attribute is (in terms of classification). Accordingly, an intuitive solution in data acquisition is to pay more attention to the attributes with larger *DE* values, so the classifier learned from a cleansed data set can possibly achieve more improvement. This intuitive solution, however, ignores the cost of the attributes, and may possibly stick to some uneconomical attributes that cost too much in comparison with their contributions. When attributes appear to have different costs, another naïve approach is to pay attention to the cheapest attributes, so given a fixed budget, this method can provide more correct (not necessarily useful) information. This approach, however, ignores the *DE* of attributes, and may possibly provide corrections with very limited contribution for inductive learning. The above observations motivate the design of a new measure which integrates the attribute cost and *DE* for data acquisition purposes. We call this measure *Economical Factor* because it shall help us explore the "economical" attributes, i.e., cheap in costs but informative for classification.

To integrate the cost and *DE* of an attribute A_i, let's assume the existence of an evidence assertion E and a hypothesis assertion H for this purpose, where H is the benefit in supporting the evidence assertion E, and $\sim H$ is the complement of the benefit (disagreement) of supporting E. With the Likelihood Ratio (*LR*) in statistics [35], defined by (3), we can quantitatively measure how much more likely the evidence E is under explanation H than under the alternative explanation $\sim$ H. A large *LR* value means that E is encouraging for H, and vice versa.

$$LR = \frac{P(E|H)}{P(E| \sim H)}. \quad (3)$$

In the context of attribute evaluation which integrates the attribute cost and the discrimination efficiency, we can find that the cost C_{A_i} and the discrimination efficiency $DE(A_i)$ of attribute A_i are actually a pair of complementary measures. We obviously expect that the evidence assertion E prefers the attribute with a cheaper cost, but a larger discrimination efficiency. Accordingly, we can take the discrimination efficiency $DE(A_i)$ as the observation assertion H, and take the cost C_{A_i}, as the $\sim$ H (the complement of H). We then propose an *Economical Factor* (*EF*) for attribute A_i, as defined by (4), where $g()$ and $f()$ represent the functions of transforming the discrimination efficiency and the cost.

$$EF_{A_i} = \frac{P(E|H)}{P(E| \sim H)} = \frac{g(DE(A_i))}{f(C_{A_i})}. \quad (4)$$

In order to define the transformation function $g()$ for *DE*, the signal noise (*S/N*) concept as a measure of efficiency of a transmission line has been adopted. The analogy of the above concept (*S/N*), from Information Theory [33], [36] is the measure between the useful and nonuseful information of the analyzed attribute, as defined by (5). Actually, we can also refer to (3) for the correctness of the prototype function defined by (5):

$$g(DE) = \frac{Useful\ Information}{Nonuseful\ Information} = \frac{UI}{NI}. \quad (5)$$

Since $UI + NI$ is equal to the Total Information TI, we can therefore transform (5) to (6), where [] defines a high-level prototype function *with regards to* the ratio between UI and NI, which will be resolved in (10).

$$g(DE) = \left[\frac{UI}{NI}\right] = \left[\frac{TI}{NI}\right] - 1. \quad (6)$$

Until now, we still do not actually know the meaning of TI/NI, so we have the following definition by taking the information-gain into consideration:

$$\Delta I = H(TI) - H(NI) = \left[\frac{H(TI)}{H(NI)}\right]. \quad (7)$$

By definition of the entropy [36], we know that

$$TI = 2^{H(TI)}; \quad NI = 2^{H(NI)}. \quad (8)$$

Therefore, we can transform (7) to (9):

$$\Delta I = \log_2\left[\frac{TI}{NI}\right], \quad (9)$$

so we have

$$2^{\Delta I} = \left[\frac{TI}{NI}\right]. \quad (10)$$

Combining (6) and (10), we will find that the prototype of the function g() is defined by (11):

$$g(DE) = 2^{\Delta I} - 1. \quad (11)$$

In (4), we define the transformation function of $f()$ by (12), to avoid (4) becoming infinite when the cost of the attribute equals to 0:

$$f(C_{A_i}) = C_{A_i} + 1. \quad (12)$$

With (4), (11), and (12), we can finally get the *EF* for attribute A_i by (13), where ΔI is the information-gain of attribute A_i [33]:

$$EF_{A_i} = \frac{2^{\Delta I} - 1}{C_{A_i} + 1}. \quad (13)$$

In our system, we use the information-gain ratio (*IR*) instead of the information-gain. With (13), there are two possibilities to introduce bias:

- If the *IR* values of all attributes are relatively small, all attributes will receive very small *EF* values because $2^{\Delta I} - 1$ is very close to 0. As a result, the costs of the attributes tend to be ignored, which leaves obscuration to assess the attributes.
- If all attributes have relatively large cost values, the attributes' *DE* tend to be ignored too because the numerator of (13) is far less than the denominator.

Our solution in solving the first problem is to normalize the *IR* values of all attributes by using (14). Actually, the numerator of (14) is the norm of the vector which takes all IR values as the elements:

$$[IR_{A_i}] = \frac{IR_{A_i}}{\sqrt{\{IR_{A_1}, IR_{A_2}, \ldots IR_{A_M}\}.\{IR_{A_1}, IR_{A_2}, \ldots IR_{A_M}\}^T}}. \quad (14)$$

To prevent the bias caused by large attribute costs, we use the average cost $\bar{C}_A = \sum_i^M C_{A_i}$ to replace the base of 2^{Δ_I}. So, the final *EF* measure for A_i is defined by (15). The higher the EF_{A_i}, the more economical A_i is, and the more likely A_i deserves a further investigation, if A_i does contain a missing value:

$$EF_{A_i} = \frac{(\bar{C}_A)^{[IR_{A_i}]} - 1}{C_{A_i} + 1}. \quad (15)$$

4 Cost-Constrained Data Acquisition

4.1 EF-Sensitive Instance Ranking for Data Acquisition (ERA)

When learning models for classification, it is obvious that not all attributes have the same impact on the system performance. The impact of an attribute is inherently determined by the correlation between the attribute and the target concept: the higher the correlation, the more important the attribute. Therefore, a data acquisition algorithm should pay attention to the instances containing missing values of the attributes, which have high correlation with the class, because those instances may possibly provide more valuable information to the learner and, therefore, enhance the system performance. In the context of cost-constrained data acquisition, we may directly use (15) to evaluate each attribute, and select the most economical instances for further investigation. However, the problem here is to determine which instances are the

most economical ones. We refer to an *EF*-sensitive instance ranking mechanism for solutions.

Given a data set D, we first calculate the *EF* value of each attribute A_i, and take the inverse of *EF* as the impact weight[1] (*IW*) for A_i. The impact value for each incomplete instance I_k, $IP(I_k)$, is then defined by (16), which is the sum of the impact weights of all attributes in I_k with missing values.

With (16), we can rank all incomplete instances by their $IP(I_k)$ values (in ascending order), and select instances from the top of the list to complete missing information. We refer to this mechanism as ERA, and will take it as a benchmark method for comparison purposes.

$$IP(I_k) = \sum_i^M IW(A_i,\ I_k);$$
$$W(A_i, I_k) = \begin{cases} 1/EF_{A_i}; & If\ A_i\ has\ a\ miss.\ value\ for\ I_k \\ 0 & otherwise. \end{cases} \tag{16}$$

In (16), we actually evaluate each instance by the sum of the inverse *EF* value from all attributes containing missing values. We use the inverse value of *EF*, which means that we prefer instances with less missing attributes. For example, assuming instance I_1 has only two missing attributes, A_1 and A_2, and instance I_2 has only one missing attribute A_1, we prefer to select I_2. Our experimental results in Section 5 will indicate that such a mechanism likely receives better performances than other alternatives, e.g., selecting I_1 instead. The reason is that preferring instances with less missing attributes can possibly provide more complete instances, given the same amount of budget.

4.2 Cost-Constrained Data Acquisition with Active ERA (AERA)

While *ERA* takes *EF* of each attribute into consideration to select economical instances, it may possibly ignore two facts: 1) an incomplete instance may already contain enough information for model construction, even though it still contains missing values, and 2) for an incomplete instance I_k, some of its missing values might be predictable by existing instances in the data set. Therefore, completing missing information for these two types of incomplete instances likely results in redundancy, and prohibits the constructed model from achieving much improvement. In this section, we propose an active ERA (AERA) algorithm that can possibly take these two facts into consideration.

The pseudocode of AERA is shown in Fig. 1. It consists of three major steps:

1. active instance selection,
2. missing value prediction, and
3. instance ranking for data acquisition.

In the first step, we borrow the essential idea of active learning to select incomplete instances that cannot be effectively classified by a benchmark classifier because those instances likely confuse the learner and deserve more attention. In the second step, for each selected incomplete instance I_k from step 1, we try to predict its missing values by using the *Attribute Prediction* mechanism (Section 4.2.2). If a missing value in I_k is predictable by others, we need not take this attribute into consideration when calculating the total *Economical Factor* value of I_k, $EF(I_k)$. After that, we conduct the third step (instance ranking) by considering that for each incomplete instance, I_k, how many attributes contain missing values, and among all these missing values, how many of them are actually predictable by others. We use all this information to calculate the *EF* value of I_k, and rank I_k based on its *EF* value. The data acquisition is conducted by selecting instances from the ranked list.

4.2.1 Active Instance Selection

The first step of AERA is to adopt the essential idea of active learning to select incomplete instances that likely confuse the learning theory, and then put more efforts on them to enhance the system performance, as shown on lines 5 to 8 of Fig. 1. For this purpose, we split the data set into n disjoint subsets. In each iteration It, we first train a benchmark classifier T_{It} from $\tilde{D}_{It}$, which is constructed by excluding the subset D_{It} from D, and then use T_{It} to evaluate each incomplete instance in D_{It}. The subset S_{It} is constructed by using the following criteria:

1. If an incomplete instance I_k in D_{It} cannot be correctly classified by T_{It}, we forward it to S_{It}.
2. If an instance I_k in D_{It} does not match any classification rule in T_{It}, we forward it to S_{It}, too.

Our method relies on the benchmark classifier trained from the noisy set for active instance selection. This is different from [17], where only complete instances were involved to train the classifier for any evaluation purpose. There are two reasons of doing so: 1) when a data set contains high level missing values, it may have very limited complete instances, therefore the theory learned from complete instances will have serious bias; and 2) for incomplete instances, the attributes without missing values can still provide valuable information to construct models.

4.2.2 Missing Value Prediction

The purpose of the missing value prediction is to evaluate whether a missing attribute value of an instance is predictable by others, and if it is predictable, we will not consider this attribute when ranking instances for data acquisition. The whole procedure is presented on lines 9 to 18 of Fig. 1. At each iteration It, given an incomplete instance I_k in S_{It}, we denote the aggregation of all attributes with missing values in I_k by MA_k (*Missing Attributes*), and the number of attributes in MA_k by K. Our objective is to evaluate that among K missing attributes in MA_k, how many of them could be predicted by using the theory T_{It}. We denote those predictable attributes in MA_k by PA_k (*Predictable Attributes*). When ranking I_k, we should exclude the attributes in PA_k from MA_k, as shown on line 16 of Fig. 1 because acquiring correct values for predictable missing attribute values will not bring much improvement to the system performance.

To evaluate whether the missing value of an attribute in MA_k is predictable or not, we adopt an *Attribute Prediction* (*AP*) mechanism, as shown in Fig. 2. Basically, Attribute Prediction uses all other attributes $A_1, \ldots A_j, \ldots A_M$ $(j \neq i)$ and the class label c to train a classifier, AP_i, for A_i (using

1. We call it impact weight, instead of weight, to emphasize on the fact that this value reveals the unified impact of an attribute (incorporating both the classification importance and the cost). In addition, the term weight is too broad and hard to characterize this particular scenario.

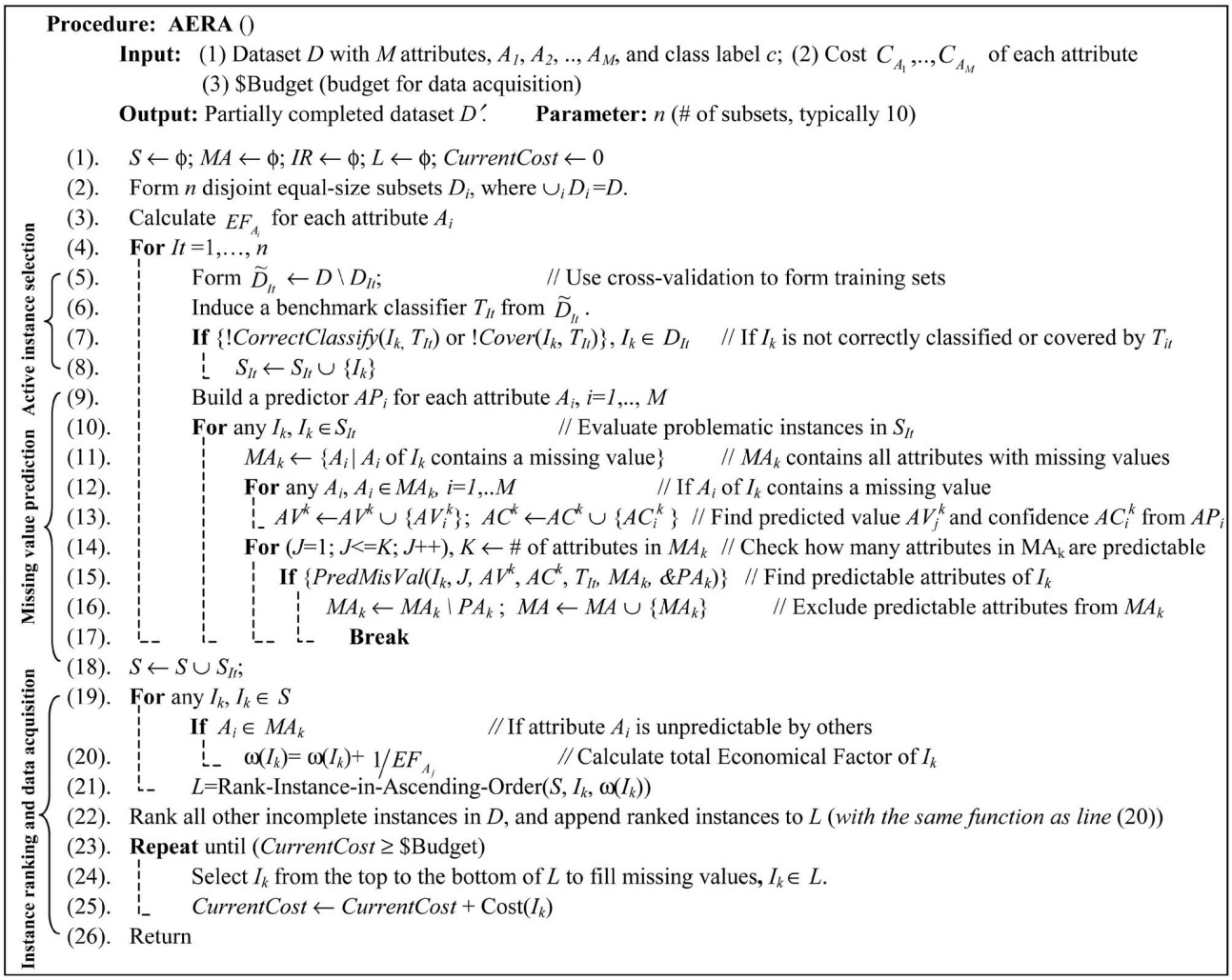


Fig. 1. Cost-constrained data acquisition (AERA).

instances in $\tilde{D}_{It}$). Given an instance I_k in S_{It}, if A_i contains a missing value, we use AP_i to predict a new value for A_i. Assuming the predicted value from AP_i is AV_i^k, we then use the benchmark classifier T_{It} to determine whether the predicted value from AP_i makes more sense: If we change the attribute value of A_i from unknown to AV_i^k (as shown on line 4 of Fig. 2, where an unknown attribute value is denoted by "?"), and I_k can be correctly classified by T_{It}, it will indicate that the change results in a better classification. We therefore conclude that the missing value of A_i is predictable by others. However, if the change still makes I_k incorrectly classified by T_{It}, we will leave A_i unchanged and try to evaluate other attributes. If the prediction from each single attribute does not conclude any predictable attribute, we will start to revise multiple attribute values at the same time. For example, change the values of two attributes A_i and A_j to AV_i^k and AV_i^k simultaneously, and then evaluate whether multiple changes make sense to T_{It}. We can iteratively execute the same procedure until a change makes I_k correctly classified by T_{It}, or none of the changes can make I_k correctly classified by T_{It} (in this case, $PA_k \leftarrow \phi$). Then, we exclude all predictable attributes from MA_k, and use the remaining attributes in MA_k to take part in the ranking of I_k.

With the above procedure, one important issue should be resolved in advance: Which attribute to select if multiple attributes are found to be predictable?

Our solution in solving this problem is to maximize the prediction confidence while locating the predictable attributes, as shown in Fig. 2. When learning AP_i for each attribute A_i, we use a classification rule algorithm, e.g., C4.5 rules [15], to learn an Attribute prediction Rule set (AR_i). Assuming the number of rules in AR_i is R_i, for each rule AR_i^r, $r = 1, \ldots R_i$, in AR_i, we evaluate its accuracy on subset $\tilde{D}_{It}$. This accuracy will indicate the confidence that AR_i^r classifies the instance. In our system, we use C4.5 rules to learn the rule set AR_i, so the accuracy value has been provided with each learned rule.

When adopting the rule set AR_i to predict the value of A_i, we use the first hit mechanism [15], which means we rank the rules in AR_i in advance, and classify instance I_k by its first coved rule in AR_i. Meanwhile, we also use the accuracy of the selected rule as the confidence (AC_i^k) of AP_i in predicting A_i of I_k.

Given an incomplete instance I_k, assume the predicted values for K attributes in MA_k are $AV_i^k, \ldots, AV_K^k$, respectively, with the confidences for each of them denoted by $AC_1^k, \ldots, AC_K^k$. We first set J to 1 to locate a predictable

Procedure: PredMisVal(I_k, J, AV^k, AC^k , T, MA_k, &PA_k)

(1). $PA_k \leftarrow \phi$; $\Omega[] \leftarrow \phi$; $ChgNum \leftarrow 0$; $Iteration \leftarrow 0$; $CV[] \leftarrow \phi$; $K \leftarrow$ # of attributes in MA_k

(2). **Repeat** until ($Iteraction > \binom{K}{J}$)

(3). **For** any J attributes, $A_{i_1},..A_{i_J} \in MA_k$; $A_{i_1} \neq .. \neq A_{i_J}$; $\{A_{i_1},..A_{i_J}\} \subset \Omega[]$

(4). Replace the missing value "?" with the predicted attribute values, $? \leftarrow AV^k_{i_1}; .., ? \leftarrow AV^k_{i_J}$

(5). **If** {$CorrectClassify$ (I_k, T)} // If the change makes I_k correctly classified by T

(6). $CV[ChgNum] = AC^k_{i_1} + ... + AC^k_{i_J}$; $\Omega[ChgNum] \leftarrow \{A_{i_1},..., A_{i_J}\}$

(7). Restore old values of $A_{i_1},...A_{i_J}$; $ChgNum$++

(8). $Iteration$++

(9). **If** {$ChgNum$!= 0} // Select the change with the maximal confidence value

(10). $PA_k \leftarrow \Omega[l]$, l=arg{max($CV[j]$), j=1,.., $ChgNum$}

(11). **Return** (1)

(12). **Else** Return (0)

Fig. 2. Missing attribute value prediction.

attribute by (17). If this procedure does not find any predictable attribute, we increase the value of J by 1 and repeat the same procedure, until we find the predictable attributes or J reaches the number of attributes in MA_k, as shown in lines 14 to 17 in Fig. 1:

$$PA_k = \{A_{i_1}, ., A_{i_J}\},$$
$$\arg_{i_1,...,i_J} \left\{ \max_{\{A_{i_1},..A_{i_J},..A_{i_J}\} \in MA_k;\, i_1 \neq .i_J . \neq i_J;\, A_{i_J} \in \forall MA_k} \left\{ \sum_l^J AC^k_{i_l} \right\} \right\}, \quad (17)$$
$$where\ CorrectClassify\ (AV^k_{i_1}, ., AV^k_{i_J}, T_{It}) = 1.$$

Validity Analysis. By switching each attribute A_i and the class c to learn an AP_i classifier for each attribute, our attribute prediction mechanism looks similar to the decision tree-based imputation method [10], where the decision tree was generated to predict the missing values for each attribute. However, there are essential differences between them. With the method in [10], it directly uses the prediction results from the decision tree trained for each attribute to fill the missing value. One can imagine that if the classification accuracy of one attribute is low, the imputation results become very unreliable (possibly even worse than a random guess) [16]. In our method, nevertheless, the prediction from each AP_i classifier just provides a guide for the benchmark classifier T_{It} to evaluate whether a change makes the classification better or not. The prediction from AP_i would not be adopted unless T_{It} agrees that the value predicted by AP_i will make the instance correctly classified. In other words, the proposed mechanism relies more on T_{It} than on any AP_i. Even if the prediction accuracy from AP_i is 100 percent, we will not take its prediction unless it gets the support from T. Therefore, low prediction accuracy from AP_i does not have much influence with our proposed algorithm.

Although our method does not crucially depend on the performance of each AP_i, we obviously prefer a high prediction accuracy from AP_i because it can increase the algorithm's reliability. Then, the question becomes how good AP_i could be with a normal data set? Actually, the performance of AP_i is determined by the correlations among attributes. If all attributes are independent or conditionally independent given the class c, the accuracy of AP_i could be very low because no attribute could be used to predict A_i. However, it has often been pointed out that this assumption is a gross oversimplification in reality, and the truth is that the correlations among attributes extensively exist [10], [37]. Instead of taking the assumption of conditional independence, we take the benefits of interactions among attributes, as well as between the attributes and class. Just as we can predict the class C by using the existing attribute values, we can turn the process around and use the class and some attributes to predict the value of another attribute. Therefore, the average accuracy of AP_i classifiers from a real-world data set usually maintains a reasonable level, as shown in Table 1.

4.2.3 Instance Ranking for Data Acquisition

We repeat procedures in Sections 4.2.1 and 4.2.2 for n times and forward all instances in S_{It} to a subset S. Our next step is to rank instances in S for data acquisition. To this end, we adopt a similar ranking mechanism to *ERA*, as shown on lines 19 to 21 in Fig. 1.

For each instance I_k in S, we first check how many attributes in I_k contain missing values; in addition, how many of these missing values are unpredictable by other instances. Then, we use the sum of the inverse *EF* value of all these (missing and unpredictable) attributes as the *EF* value of the instance I_k, as defined by (18):

$$EF(I_k) = \sum_i^M \left\{ \frac{1}{EF_{A_i}} \Big|, A_i \text{ is missing and unpredictable} \right\}. \quad (18)$$

After we calculate the *EF* value for each instance in S, we rank all instances in ascending order and put instances into the list L. In addition, for other incomplete instances which do not belong to S, we also use (18) to rank them (but without considering whether an attribute is predictable or not). Then, we append the ranked instances to L (because these instances are likely less important than the instances in S, we would like to give them less priority). Finally, all incomplete instances are ranked.

TABLE 1
Missing Attribute Value Prediction Results

Dataset	x·100%	PR	PA	PA_1	PA_2	PA_3	PA_4	PA_5	PA_6	PA_7	PA_8	PA_9
Auto-mpg	10%	15.24	45.87	25.07	94.41	81.67	45.28	44.68	23.33	6.67	N/A	N/A
	30%	7.87	41.68	21.33	91.54	73.54	40.37	45.71	15.12	4.16	N/A	N/A
	50%	3.09	31.88	15.22	87.99	65.14	22.00	24.34	7.39	1.08	N/A	N/A
Car	10%	12.79	55.68	55.74	52.75	31.4	59.14	60.02	75.06	N/A	N/A	N/A
	30%	11.03	50.54	53.24	47.88	26.83	57.33	53.61	64.34	N/A	N/A	N/A
	50%	8.44	36.61	28.66	32.47	11.70	59.27	24.49	63.07	N/A	N/A	N/A
LED24	10%	9.79	33.26	100	100	100	100	100	100	100	0	15.02
	30%	6.94	38.92	100	100	100	99.78	100	100	100	9.64	8.34
	50%	4.45	33.41	100	99.27	100	98.36	100	100	98.95	6.67	3.33
Soybean	10%	9.91	55.38	35.68	41.67	87.46	55.33	23.35	18.99	44.87	48.42	12.78
	30%	7.41	65.56	31.32	53.24	60.07	57.25	27.22	9.37	44.72	44.39	7.17
	50%	5.47	50.23	15.69	23.33	51.67	44.55	21.62	4.32	34.68	9.27	5.88
Tictactoe	10%	17.49	80.12	80.99	78.65	83.26	72.95	84.98	76.48	83.27	78.80	81.67
	30%	9.47	63.42	68.44	60.12	69.91	56.26	70.79	52.57	68.34	52.73	71.66
	50%	5.92	42.06	44.25	37.08	47.49	32.91	69.54	32.08	46.67	21.33	47.16

Given a user-specified budget, AERA starts from the top to the bottom of the list L and selects instances to fill the missing values, as shown on lines 23 to 25 in Fig. 1. AERA keeps selecting instances from the list L, until the budget has been filled, and then returns partially completed data set D'.

5 Experimental Results

5.1 The Objective and Evaluation Criteria

The objective of our experimental comparisons is to evaluate the performances of our proposed efforts in acquiring missing attribute values in cost bounded environments. Our assumption is that each data set comes with a cost matrix which specifies the price for acquiring a missing value for each single attribute. Meanwhile, we also assume that when acquiring a missing value, the user has the power of completing any particular missing value, as long as he/she is willing to pay the price and the total cost is bounded by a previously given number. We make this strong assumption for objective evaluation purposes, although in reality, many factors may actually impact the procedure and make some missing values unable to restore. When comparing different approaches, the major evaluation criterion is the classification accuracy improvement from the models built from the data sets that have been processed by different data acquisition methods, given the same data sets and the same cost matrix and budget.

In the following sections, we will conduct extensive studies to assess the performances of the proposed efforts, in comparison with several benchmark methods, which include the missing value prediction accuracy, the data acquisition results of various missing value levels, different attribute costs, different cost budgets, and the results from one real-world case study.

5.2 Experiment Setting

The majority of our experiments use C4.5, a program for inducing decision trees [16]. To construct the benchmark classifier T_{It} and AP_i classifiers, C4.5rules [16] is adopted in our system. We report and analyze our results on 12 data sets from the UCI data repository [32]. For each numerical attribute, we will discretize it into 10 levels in advance (using the k-means clustering algorithm [38]).

We did not intentionally select those data sets in UCI which originally come with missing values because even if they do contain missing values, we still do not know the correct value to complete the missing information. So, we propose a random missing value corruption model to systematically study the performance of the proposed data acquisition mechanisms. With this corruption model, when the user specifies a corruption level $x \cdot 100\%$, we will introduce missing values to all attributes, with the value of each attribute having $x \cdot 100\%$ of chance to be hidden (corrupted as missing values). If the original value of the attribute is unknown (it happens when the data set comes with missing values), we still set it as unknown. In our system, we set $x \in [0.1, 0.5]$.

For each experiment, we execute 10 times 3-fold cross-validation and use the average as the final result. In each run, the data set is randomly (with a proportional partitioning scheme) divided into a training set and a test set, and we apply the above corruption model on the training set. The learner first builds a classifier (C_1) from the corrupted data set, then we use the data acquisition schemes to complete a certain amount of data.[2] After that, the leaner builds another classifier (C_2). The comparison of system performance is made by comparing the classification accuracy between C_1 and C_2 (on the same test set).

To assign costs for attributes, we adopt a random mechanism (unless otherwise specified; e.g., for the Heart

2. With manual corruption schemes, we actually know the original value of the missing attribute, so we can complete an incomplete instance with 100 percent accuracy. If the original data set contains a missing value for an attribute (e.g., Soybean and Vote), such a procedure will still provide a missing value for the attribute because we do not actually know the correct attribute value.

disease data set, we use the costs from the original data set). Given a maximum attribute cost *MaxAttCst*, we randomly select a cost value within the constraint [1, *MaxAttCst*] for each attribute. The cost values for attributes are generated at the beginning of each trial of 10 time cross-validation. When the system picks one instance, say I_k, for users to complete the missing information, we assume that users can provide correct information for all attributes in I_k with missing values. We take this assumption because from the data acquisition point of view, once the user has started to investigate one instance, he/she has already gotten the background knowledge of I_k, and it is convenient (also reasonable) for the users to complete the missing values of I_k as much as they can (because if the user only picks some attributes to fix, he/she may reexamine the instance again at the later stage). In our system, we take this assumption for all data acquisition algorithms.

For comparison purposes, we also consider several naïve data acquisition solutions and implement one existing algorithm AVID [27], and use them as the benchmark to evaluate the performance of the proposed methods.

5.2.1 Random Data Acquisition (Random)

Random data acquisition randomly selects incomplete instances for data acquisition until the user-specified budget has been filled.

5.2.2 The Most/Least Expensive (MstExp/LstExp)

With these two mechanisms, the system will rank all incomplete instances in D by their total cost. The MstExp always tries to fix the instance with the most expensive cost at first, and the LstExp, inversely, always prefers the instance with the least expensive cost. Both mechanisms discard the importance of the attributes. The intuitive sense is that MstExp can finish the job by checking the least number of examples, and is suitable for users who want to finish their job quickly. LstExp, on the other hand, examines most examples and is suitable for users who do not care about the time cost.

5.2.3 The Most Informative (MstInf)

With this mechanism, the system first calculates the Information-gain Ratio (*IR*) of each attribute. For each incomplete instance I_k, MstInf uses the sum of *IR* of all attributes with missing values as the impact value of I_k. MstInf selects the instances with the highest impact values until the budget has been filled. Obviously, MstInf disregards attribute costs but considers the importance of the attributes only.

5.2.4 Acquisition Based on Various Imputation Data (AVID)

For a better comparison, we also implement an existing data acquisition algorithm, AVID [27]. The essential idea of AVID is to put priority on the missing attribute with the largest variance of the imputation results. Given one instance, I_k, if attribute A_i of I_k contains a missing value, AVID first introduces B imputation methods to predict the value of A_i, and then calculate the variance from the imputation methods. The total score of instance I_k is based on the average variance of all attributes with missing values, as defined by (19). The larger the score, the more important I_k is for data acquisition. When implementing AVID, we use three imputation algorithms, i.e., $B = 3$, which are the most common attribute values [12], the Bayesian formalism [13], and the decision rule prediction [10]. The final score for I_k is calculated by (19):

$$S(I_k) = \sum_{i=1}^{M} \left\{ \left(1 - \frac{Max(B_{A_i})}{B}\right) \middle| A_i \text{ has a missing value} \right\}, \tag{19}$$

where B_{A_i} represents the imputation results for attribute A_i (with a missing value), and $Max(B_{A_i})$ denotes the number of imputation methods with the same results. Obviously, if all imputation results for A_i are the same, $S(I_k)$ equals to 0. We rank all incomplete instance by their $S(I_k)$ in descending order, and select the instances from the top to the bottom of the list for data acquisition until the budget has been satisfied.

With the above five mechanisms and the ERA method introduced in Section 4.1, we actually have six benchmark algorithms for comparison purposes.

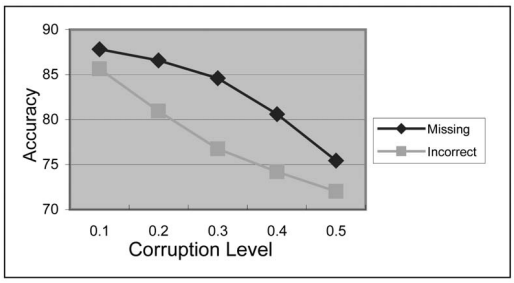
5.3 Missing Attribute Value Prediction Results

AERA has integrated a missing value prediction mechanism to evaluate whether a missing value of an instance is predictable by others. We need to assess the performance of such a prediction procedure because if the prediction accuracy is very low, it may provide wrong information, and AERA will falsely ignore the missing values from important attributes. Therefore, for each attribute, say A_i, we will evaluate its *Prediction Accuracy* (*PA*) and *Prediction Recall* (*PR*), as defined by (20), where n_i, p_i, and d_i represent the number of actual corrupted missing values, the number of correctly predicted values, and the number of predicted values in A_i, respectively. PA_i and PR_i are the accuracy and recall for A_i, and *PA* and *PR* are the average accuracy and recall of all attributes. Actually, we define the accuracy as that among all predictions from the algorithm, how many of them are exactly the same as the original values (before the missing value corruption). And, the recall is defined as that among all actually corrupted missing values (for each attribute), how many of them are correctly detected by the algorithm.

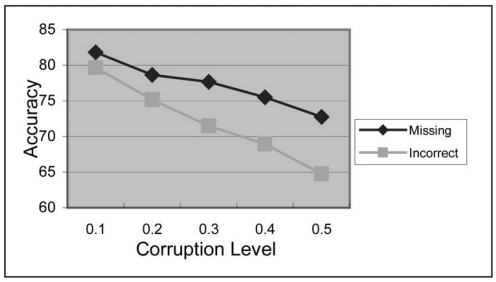
Table 1 reports the results from several representative data sets (we only report the results from the first nine attributes, where N/A means the data sets do not have so many attributes).

$$\begin{aligned} PA &= \sum_{i=1}^{M} PA_i, \quad PA_i = \frac{p_i}{d_i}; \\ PR &= \sum_{i=1}^{M} PR_i, \quad PR_i = \frac{p_i}{n_i}. \end{aligned} \tag{20}$$

As we can see from Table 1, when the corruption level ($x \cdot 100\%$) is relatively low, the average prediction accuracies from all data sets are reasonably good. When the corruption level goes higher, the accuracies likely decrease. This makes sense because with more missing values introduced, both learned attribute classifiers AP_i and the benchmark classifier T_{It} become worse. On average, about half of the predictions are correct, which implies that the



(a)



(b)

Fig. 3. Classification accuracy comparison between missing values and incorrect values. (a) Car data set and (b) Tictactoe data set.

proposed algorithm looks promising in predicting some missing attribute values. However, for some attributes (e.g., attributes eight to 24 of LED24, and attribute 7 of Auto-mpg), the prediction accuracies are very low, regardless of the corruption level. Further analysis indicates that there are cases that some attributes in the data set simply cannot be predicted by others. For example, the values in attributes 8 to 24 of LED24 are randomly generated, which leaves no way for a high prediction accuracy. But, since a data set is likely constructed with lots of interactions among attributes [10], having some independent attributes in the data sets does not impact too much on our algorithm.

As we can see from the third column of Table 1, the overall recall of the prediction is relatively low (on average, less than 10 percent), which means that the algorithm can only handle less than 10 percent of missing attribute values. Simply looking at this value might raise an argument against the efficiency of the algorithm. Nevertheless, this is actually the merit of our algorithm. Instead of predicting missing values for all incomplete instances (like most imputation methods do), we only select and predict the missing values for the most problematic incomplete instances. In addition, our prediction mechanism depends on the trained benchmark classifier T_{It}, where those less important attributes, e.g., the random attributes 8 to 24 in LED24, are less likely to show up in theory T_{It}, so our prediction mechanism will make less prediction for these attributes. Our experimental results in the following sections will indicate that our method contributes very significantly for data acquisition mechanisms, especially when the user's budget is very small.

The results from some attributes (e.g., in LED24 and Tictactoe) likely suggest that when the missing values in a data set are very limited, we may use the predicted attribute values to change the data set (notice that AERA does not do so). However, the risk is that incorrectly predicted attribute values may harm the system. So, we compare the same levels of missing and incorrect attribute values to check which one is more harmful. Given one data set, we first introduce a certain level of missing values, and acquire a classifier C_1. Then, we change each of the manually corrupted missing attribute value with a random attribute value, and learn a classifier C_2. We execute experiments on Car and Tictactoe at five corruption levels and report the results in Fig. 3, where "Missing" and "Incorrect" represent the accuracy of C_1 and C_2, respectively. As we can see, given the same corruption level, the incorrect attribute values do much more harm to the system than missing values. This clearly suggests that instead of bringing benefits, adopting "automatic" imputation methods to change the missing attribute values may actually bring more negative impacts. So, AERA does not use any predicted values to change the original data set.

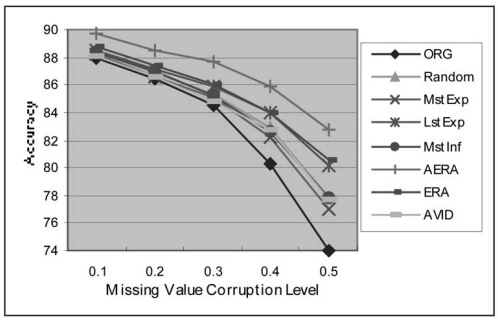


Fig. 4. Classification accuracy comparison at various corruption levels (Car data set, $\alpha = 0.1$, $\mathrm{MaxAttCst} = 20$).

5.4 Experiments with Various Corruption Levels

In this section, we evaluate the performances of different algorithms on data sets with different levels of missing values ($x \cdot 100\%$). For each experiment, we need to specify the maximum attribute cost MaxAttCst to generate attribute costs (Section 5.1). Based on the generated attribute costs and the number of attributes with missing values, we can calculate the total cost we will have to pay to complete all missing information in the data set, denoted by *TotalCst*. In the context of cost-constrained environments, users will specify a budget for data acquisition, which is $\alpha \cdot 100\%$ of the *TotalCst*. In all experiments in this section, we set $\mathrm{MaxAttCst} = 20$ and $\alpha = 0.1$. (We believe both of these two values are representative in reality and, in Sections 5.5 and 5.6, we will evaluate the impacts of MaxAttCst and α on the system performance.) Then, we learn classifiers from the data sets that have been processed by different data acquisition mechanisms, and report their accuracy in Fig. 4 and Table 2. Fig. 4 provides detailed results from the Car data set, where the *x*-axis represents the corruption level and the *y*-axis is the classification accuracy. ORG means the accuracy of the classifier learned from the corrupted data set (without any data acquisition), and each of the other curves represents the results from one data acquisition method (including six benchmarks and one proposed solution).

From Fig. 4, we can find that with the increase of the corruption levels, all methods suffer from the decrease of

TABLE 2
Classification Accuracy with Various Corruption Levels ($\alpha = 0.1$, $\mathrm{MaxAttCst} = 20$)

Dataset	x·100%	Random	LstExp	AVID	ERA	AERA	Dataset	x·100%	Random	LstExp	AVID	ERA	AERA
Auto-mpg	10%	66.92	67.11	66.38	67.49	**68.91**	Tictactoe	10 %	81.57	81.24	81..51	81.78	**83.65**
	30%	66.71	66.97	66.59	67.87	**68.40**		30 %	78.58	78.86	78.61	78.33	**79.98**
	50%	65.85	65.99	66.07	66.51	**67.79**		50 %	75.25	75.34	73.94	75.51	**76.69**
Krvskp	10%	97.43	97.91	97.66	97.78	**98.85**	Vote	10%	93.63	93.68	93.83	93.74	**94.21**
	30%	96.18	96.15	95.71	96.43	**97.45**		30%	92.55	92.72	92.66	92.91	**93.89**
	50%	94.22	95.14	94.86	94.89	**96.83**		50%	91.37	91.46	91.21	91.66	**92.68**
Led24	10%	96.83	97.51	97.13	97.08	**99.52**	Wbreastc	10%	92.80	92.23	92.56	92.78	**93.74**
	30%	94.01	95.04	94.81	95.21	**97.79**		30%	91.45	91.89	91.34	91.62	**92.03**
	50%	94.14	95.31	94.61	95.41	**96.78**		50%	91.69	91.42	90.12	91.08	**91.75**
Lympho	10%	73.29	72.44	72.69	73.13	**74.56**	Wine	10%	80.78	80.96	81.65	80.59	**82.89**
	30%	71.52	71.59	71.36	73.42	**74.89**		30%	78.16	79.61	78.85	80.07	**81.11**
	50%	70.22	69.54	70.28	71.01	**72.97**		50%	77.03	77.59	75.79	77.06	**79.84**
Soybean	10 %	84.37	84.78	84.99	84.89	**85.98**	Zoo	10%	87.65	87.68	86.79	87.33	**89.57**
	30 %	82.61	81.86	83.17	82.34	**83.89**		30%	86.19	85.34	85.70	85.08	**86.84**
	50 %	78.84	77.72	80.08	78.97	**81.41**		50%	80.40	80.75	80.72	81.54	**83.79**

the classification accuracy because more missing values have been introduced and the model generated from the data has been corrupted. With data acquisition mechanisms, we can improve the performance of the classifier learned from the processed data sets. Even random data acquisition can achieve a good performance, especially when the corruption level is relatively high. When comparing different data acquisition algorithms, we can find that AERA is always better than the other methods, and ERA and LstExp appear to be the second best methods (with ERA being slightly better than LstExp). If we rank all approaches in descending order of the performance, we will get the list from AERA, ERA, LstExp, MstInf, Random, AVID to MstExp. When comparing LstExp and MstExp, we can find that LstExp is much better than MstExp (this phenomenon has been found from most data sets). We believe the reason is that given the same budget, LstExp provides the data set with more complete instances (in comparison with MstExp), so the generated model can acquire a better performance. When comparing Random with all other approaches that ignore the attribute costs (MstInf and AVID), we can find that their performances are comparable, and it is hard to conclude which one is clearly the best. This indicates that in the context of cost-constrained environments, most existing data acquisition models tend to fail, and their performances are hardly better than random selection. This conclusion becomes especially clear in the experiment of the next section, where the MaxAttCost becomes relatively large, e.g., 100. With the AVID mechanism, we can find that its performance is far less than promising. The problem of AVID is that it ignores the attribute importance and heavily relies on the imputation results (which are also questionable). Therefore, its performance is almost the same as Random, even if we set the MaxAttCost to a very small value, e.g., 2, as we will discuss in the next section.

With cost-constrained data acquisition, we focus on the attributes rather than each instance. So, given a fixed budget, different algorithms may need to process different numbers of incomplete instances. An intuitive sense is that an "optimal" data acquisition model should process the least number of instances but acquire the best accuracy improvement, given a fixed budget. To make a comparison in this regard, we compare the percentages of processed instances by different algorithms, and provide the results in Fig. 5, where the x-axis represents the corruption level and the y-axis is the percentage of processed instances (in comparison with the total number of instances in the data set). As we can see, LstExp receives the highest percentage value, which means that LstExp has to process the most instances, given a fixed budget. In another word, LstExp has the lowest efficiency, if the user takes the number of processed instances into consideration. From the results in Figs. 4 and 5, it is not hard for us to conclude that AERA is the best solution if we take both the system improvement and the percentage of processed instances into consideration.

In Table 2, we summarize the results from 10 benchmark data sets, where the second column represents the corruption levels ($x \cdot 100\%$), and all other columns from 3 to 7 report the results from different algorithms (due to space restrictions, we ignore the performance of MstExp and MstInf). The text in bold indicates the result with the maximum value, and the text in the highlighted region is the second maximum value.

As we can see, from all 10 data sets, AERA is significantly better than all other methods. On average, AERA acquires 2-3 percent more improvement than others. When comparing four benchmark methods (Random,

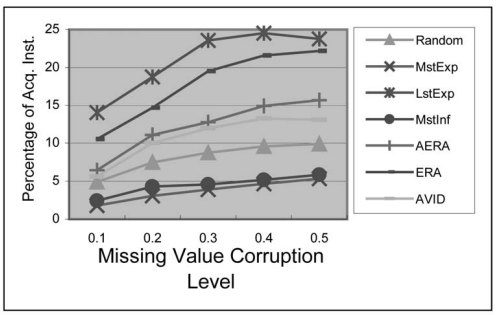


Fig. 5. Percentages of processed instances at various corruption levels (Car data set, $\alpha = 0.1$, $\mathrm{MaxAttCst} = 20$).

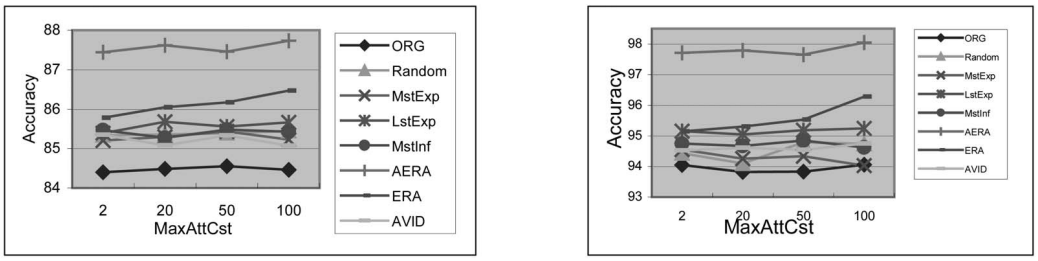


Fig. 6. Classifiction accuracy comparison with different attribute cist-ratios ($\alpha = 0.1$, $x = 0.3$). (a) Car data set and (b) LED24 data set.

LstExp, AVID, and ERA), we can find that ERA receives a relatively better performance, and can beat all other methods more often (with a 14/30 possibility). This complies with our intuitive assumption that the *EF* measure, which combines the attribute cost and importance, appears to be promising for data acquisition purposes. Nevertheless, as we can see, even if ERA does outperform others, the improvement is very limited, and only if we integrate the *EF* measure with active learning and attribute prediction, a significant amount of improvement could be achieved.

5.5 Experiments with Various MaxAttCst Values

In the above section, we have fixed the maximum attribute value ($\mathrm{MaxAttCst} = 20$), so the attribute cost-ratio between the most and the least expensive attributes is no larger than 20. In this section, we evaluate the impacts of different attribute cost-ratios (the cost ratio between the most and the least expensive attributes) on the system performances, and this study shall help us explore the merits and disadvantages of different data acquisition algorithms in the context of cost-sensitive environments. For this purpose, we set the MaxAccCst to four values (2, 20, 50, and 100). In each experiment, we will fix the budget to be 10 percent ($\alpha = 0.1$) of the total cost TotalCst, and the missing value corruption levels $x \cdot 100\%$ is set to 30 percent. In Fig. 6, we report the results from two data sets (Car and LED24), where the *x*-axis represents MaxAttCst, and the *y*-axis is the classification accuracy after the data acquisition.

The experimental results from Fig. 6 indicate that AERA outperforms all other six benchmark approaches, regardless of the value of MaxAttCst. It implies that AERA is reliable in real-world environments (where the attribute cost-ratio could vary from several to over hundreds). With the increase of the MaxAttCst, we noticed that the performances from five benchmark approaches (Random, MstExp, LstExp, MstInf, and AVID) likely keep constant, and their results are similar to the results of random selection. This is understandable because these mechanisms do not have any solution to handle variances among the attribute costs for effective data acquisition, the change of the attribute cost-ratio does not impact to these methods too much. One interesting result is that when the MaxAttCst is relative large, e.g., 100, the performance of Random is actually better than most other benchmark algorithms. As shown in Fig. 6, when MaxAttCst=100, Random is better than MstInf, MstExp, and AVID. This embarrassing observation indicates that in cost-constrained environments, if the attribute cost varies significantly, existing data acquisition methods will inevitably fail. With ERA, however, we can find a clear ascending trend, with the increase of the MaxAttCst. The higher the MaxAttCst value, the better the performance of ERA is, and the closer ERA and AERA are. The reason is that ERA uses the *EF* measure to select economical attributes, and the merit of *EF* becomes more significant with the increase of the attribute cost-ratios. With AERA, we do not find such a strong trend of increase (although we do notice a small amount of changes) because AERA has already optimized the system performance, and its result is very close to the up ceiling (see next section for details). So, it is hard for AERA to exhibit a strong performance boost with the increase of the MaxAttCst.

5.6 Experiments with Various Budget Values

In the experiments above, we have fixed the budget to be 10 percent of the total cost of the database ($\alpha = 0.1$). Nevertheless, a good data acquisition algorithm should work well with different α values, especially when the α value is small because users always prefer that a significant performance could be achieved with a very small amount of costs, and there is clearly no need to adopt any proposed mechanism if the user decides to fill all incomplete information in the data set. So, we set α to different values with $\alpha \in [0.1, 0.5]$, and the corruption levels are set to three values 0.1, 0.3, and 0.5. Fig. 7 presents detailed results from the Car data set, where the *x*-axis represents the value of α, and the *y*-axis denotes the classification accuracy.

For better comparisons, we introduce the concept of "*up ceiling*" of data acquisition in evaluating different algorithms. When conducting data acquisition, we assume that if we complete all missing information in the data set, we can get the best classification accuracy. So, we use this accuracy as the up ceiling of data acquisition, denoted by UpCeiling in Fig. 7. Note that although all results are theoretically bounded by this ceiling, there are many exceptions where the results are not well bounded, because we normally use divide and conquer learning algorithms (e.g., C4.5), where providing good information does not necessarily result in better results or vice versa, as shown in the Heart data set in Fig. 8.

The results from Fig. 7 indicate that when the budget (α) increases, the results from all data acquisition schemes get better. This does not surprise us, because when data acquisition fills more incomplete instances, we actually

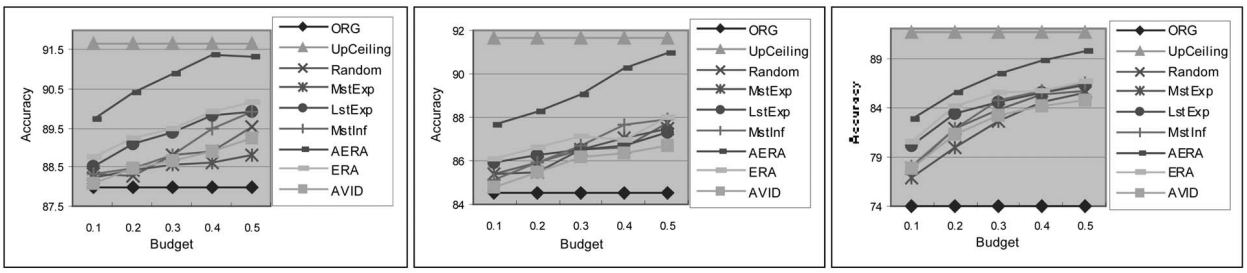


Fig. 7. Classification accuracy comparison with different user-specified budgets (Car data set). (a) $x = 0.1$, TotalCst = 4,722.5. (b) $x = 0.3$, TotalCst = 16,104.1. (c) $x = 0.5$, TotalCst = 33,170.4.

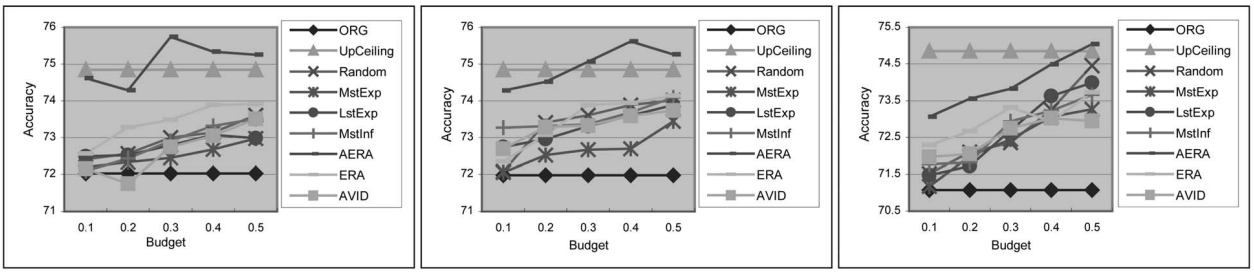


Fig. 8. A case study on the Heart disease data set with real-world attribute costs. (a) $x = 0.1$, TotalCst = 9,670.3. (b) $x = 0.3$, TotalCst = 34,868.8. (c) $x = 0.5$, TotalCst = 51,021.7.

provide more valuable information to the data set, therefore better performances could be achieved. Fig. 7 also shows that AERA achieves the best results, regardless of the budget value. When comparing results from different corruption levels, we can find that the smaller the x value, the earlier AERA hits the up ceiling, if we increase the budget (α). As we can see, when $x = 0.1$, AERA almost hits the ceiling by setting the value of α to 0.4. When $x = 0.5$, AERA is still significantly less than the ceiling even if we set α to 0.5. The reason is that with a small corruption level, we can acquire an accurate benchmark classifier T_{It} for active instance selection and attribute prediction. Therefore, AERA could achieve better results.

The results in Fig. 7 also indicate that the higher the corruption level, the more likely data acquisition can achieve a significant improvement. As shown in Fig. 7a and Fig. 7c, when introducing 10 percent missing values to each attribute, the results from all benchmark mechanisms (excluding AERA) are close to the original data set. On the other hand, when each attribute has 50 percent missing values, all benchmark algorithms are significantly better than the original data set. The reason is that when a data set contains heavy missing values, the theory learned from the data set is seriously corrupted, and putting a small amount of correct information to the data set can restore the system performance significantly. This indicates the needs and the merits of data acquisition in real-world applications, especially when the data set contains heavy missing values. The above observation also shows the merit and advantage of AERA, especially when the budget is very limited, e.g., $\alpha = 0.1$. As we can see from Fig. 7a, when $x = 0.1$ and $\alpha = 0.1$, AERA can still receive a significant improvement. This is very interesting because it indicates that AERA is particularly useful when the user has a very limited budget.

In Table 3, we summarize the results from 10 benchmark data sets, where the second column represents the user-specified budget value (α), and all other columns from 2 to 6 report the results from different algorithms (due to space restrictions, we only report the results from a representative corruption level (30 percent)). The text in bold indicates the result with the maximum value, and the text in the highlighted region represents the second maximum value. Again, all experimental results demonstrate the effectiveness of AERA in comparison with all benchmark algorithms, where AERA is significantly (and exclusively) better, regardless of the user specified budget values.

5.7 A Case Study on Real-World Attribute Costs

All experiments above are based on a random attribute cost generation. In this section, we perform a case study on a real-word data set, the Heart disease data set [32] from the UCI data repository, where attribute costs come with the original data set [31], as shown in Table 4. We introduce three corruption levels, $x = 0.1$, 0.3 and 0.5, and report the results in Fig. 8, where the *x*-axis represents the budget α, and the *y*-axis denotes the classification accuracy.

As we can see from Fig. 8, adopting the proposed cost-constrained data acquisition can also achieve a good performance from a real-world cost scenario, where given the same amount of predefined budget, AERA can improve the system performance remarkably, in comparison with all benchmark solutions. However, we notice that with the Heart disease data set, it is quite often that AERA does not comply with the up ceiling bound of data acquisition. Even if we increase the number of cross validation, the same phenomenon still exists. It seems to us that the adopted learning algorithm (C4.5) tends to overfit the training set, if we complete all missing information of the data set. Another possible reason is that we have adopted the *k*-means [38] based discretization method to discretize the

TABLE 3
Classification Accuracy with Various Budget Values ($x = 0.3$, MaxAttCst = 20)

Dataset	α100%	Random	LstExp	AVID	ERA	AERA
Auto-mpg	10%	66.71	66.97	66.59	67.87	**68.40**
	30%	67.22	68.26	67.49	67.70	**69.15**
	50%	67.56	68.34	67.39	67.91	**69.87**
Krvskp	10%	96.18	96.15	95.71	96.43	**97.45**
	30%	96.93	97.32	96.95	97.34	**98.53**
	50%	97.90	98.09	97.71	**98.52**	98.41
Led24	10%	94.09	95.04	94.91	95.21	**97.79**
	30%	96.17	97.03	96.12	97.52	**99.93**
	50%	98.05	98.73	97.48	99.44	**100.00**
lympho	10%	71.52	71.59	71.36	73.42	**74.89**
	30%	71.77	71.64	73.47	72.87	**75.67**
	50%	72.35	72.18	73.66	73.76	**75.49**
Soybean	10 %	82.61	81.86	83.17	82.34	**83.89**
	30 %	83.42	82.96	83.96	83.47	**85.31**
	50 %	84.04	83.37	83.82	84.34	**86.63**

Dataset	α100%	Random	LstExp	AVID	ERA	AERA
Tictactoe	10 %	78.58	78.86	78.61	78.33	**79.98**
	30 %	79.82	79.97	79.72	80.47	**82.26**
	50 %	80.17	80.29	80.24	81.28	**83.59**
Vote	10%	92.55	92.72	92.66	92.91	**93.89**
	30%	93.03	93.18	93.11	92.78	**93.83**
	50%	93.12	93.29	93.42	93.49	**94.24**
Wbreastc	10%	91.45	91.89	91.34	91.62	**92.03**
	30%	91.68	92.15	90.78	91.89	**93.44**
	50%	91.56	92.33	90.95	92.06	**92.73**
wine	10%	78.16	79.61	78.85	80.07	**81.11**
	30%	78.63	80.01	80.32	80.51	**82.56**
	50%	80.14	81.37	80.52	81.34	**83.78**
zoo	10%	86.19	85.34	85.70	85.08	**86.84**
	30%	86.26	85.75	85.34	86.54	**88.35**
	50%	87.12	87.38	86.90	87.25	**88.97**

TABLE 4
Attribute Costs for the Heart Disease Data Set

Attribute & Description	Cost	Attribute & Description	Cost
Att1: patient's age	\$1	Att7: electrocardiograph	\$15.5
Att2: patient's gender	\$1	Att8: maximum heart rate	\$102.9
Att3: chest pain type	\$1	Att9: exercise induced angina	\$87.3
Att4: blood pressure	\$1	Att10: ST depression induced by exercise	\$87.3
Att5: serum cholesterol	\$7.3	Att11: slope of peak exercise ST	\$87.3
Att6: fasting blood sugar	\$5.2	Att12: number of major vessels coloured by fluoroscopy	\$100.9
Att13: thal, 3=normal; 6= fixed defect; 7=reversible defect			\$102.9

numerical attributes (into 10 levels), which might have lost some information, and therefore caused this problem. Despite these reasons, we have found that even if the overfiting does exist, AERA can still explore the economical incomplete instances to enhance the data quality, and improve the system performances or serve other data mining purposes.

6 Conclusions

In this paper, we have proposed a cost-constrained data acquisition mechanism to recommend incomplete instances in a data set for information completion, where the costs of fixing the missing information of different attributes are inherently different, and the data acquisition is bounded by a user-specified budget. The essential ideas that motivate the design of our algorithm, and also the novel features that distinguish our work from existing approaches, include three components:

1. We seamlessly combine the cost and the discrimination efficiency of each attribute into a unique Economical Factor to explore the economical attributes (informative but cheap) for effective data acquisition;
2. Use the basic idea of active learning to select incomplete instances which likely confuse the learning theory to enhance the system performance; and
3. Adopt a missing attribute value prediction to avoid introducing redundancy from information completion.

The experimental results from 12 real-world data sets have demonstrated the effectiveness of the proposed approach. Our experimental results concluded that in the context that fixing attributes comes with different prices, the existing data acquisition mechanisms become less efficient and can totally fail, especially when the attribute cost-ratio (the cost ratio between the most and the least expensive attributes) becomes relatively large. With the proposed AERA algorithm, which integrates the attribute cost and discrimination efficiency, we can significantly improve the accuracy of the models built from the processed data set, compared with several benchmark approaches.

References

[1] M. Berry and G. Linoff, *Mastering Data Mining.* Wiley, 1999.
[2] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2001.
[3] D. Pyle, *Data Preparation for Data Mining.* Morgan Kauffman, 1999.
[4] T. Redman, *Data Quality for the Information Age.* Artech House, 1996.
[5] J. Quinlan, "Unknown Attribute Values in Induction," *Proc. Sixth Int'l Conf. Machine Learning Workshop,* pp. 164-168, 1989.
[6] R. Greiner, A. Grove, and A. Kogan, "Knowing What Doesn't Matter: Exploiting the Omission of Irrelevant Data," *Artificial Intelligence,* vol. 97, nos. 1-2, Dec. 1997.
[7] D. Schuurmans and R. Greiner, *Learning to Classify Incomplete Examples, In Computational Learning Theory and Natural Learning Systems: Making Learning Systems Practical.* MIT Press, 1996.

[8] N. Friedman, "Learning Belief Networks in the Presence of Missing Values and Hidden Variables," *Proc. Int'l Conf. Machine Learning,* pp. 125-133, 1997.
[9] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees.* Wadsworth & Brooks, 1984.
[10] A. Shapiro, *Structured Induction in Expert Systems.* Addison-Wesley, 1987.
[11] R. Little and D. Rubin, *Statistical Analysis with Missing Data.* New York: Wiley, 1987.
[12] P. Clark and T. Niblett, "The CN2 Induction Algorithm," *Machine Learning,* vol. 3, no. 4, pp. 261-283, 1989.
[13] I. Kononenko, I. Bratko, and E. Roskar, "Experiments in Automatic Learning of Medical Diagnostic Rules," technical report, Jozef Stefan Inst., Ljubljana, Yugoslavia, 1984.
[14] S. Tseng, K. Wang, and C. Lee, "A Pre-Processing Method to Deal with Missing Values by Integrating Clustering and Regression Techniques," *Applied Artificial Intelligence,* vol. 17, nos. 5-6, pp. 535-544, 2003.
[15] J. Quinlan, *C4. 5: Programs for Machine Learning.* San Mateo, Calif.: Morgan Kaufmann, 1993.
[16] J. Quinlan, "Induction of Decision Trees," *Machine Learning,* vol. 1, pp. 81-106, 1986.
[17] D. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 2, pp. 408-421, 1972.
[18] D. Aha, D. Kibler, and M. Albert, "Instance-Based Learning Algorithms," *Machine Learning,* vol. 6, no. 1, pp. 37-66, 1991.
[19] D. Wilson and T. Martinez, "Reduction Techniques for Examplar-Based Learning Algorithms," *Machine Learning,* vol. 38, no. 3, pp. 257-268, 2000.
[20] P. Winston, "Learning Structural Descriptions from Examples." *The Psychology of Computer Vision,* New York: McGraw-Hill, 1975.
[21] F. Provost, D. Jensen, and T. Oates, "Efficient Progressive Sampling," *Proc. Fifth ACM SIGKDD,* pp. 23-32, 1999.
[22] D. Lewis and J. Catlett, "Heterogeneous Uncertainty Sampling for Supervised Learning," *Proc. 11th Int'l Conf. Machine Learning,* pp. 148-156, 1994.
[23] D. Cohn, L. Atlas, and R. Ladner, "Improving Generalization with Active Learning," *Machine Learning,* vol. 15, pp. 201-221, 1994.
[24] H. Seung, M. Opper, and H. Sompolinsky, "Query by Committee," *Proc. ACM Workshop Computational Learning Theory,* 1992.
[25] D. Mackay, "Information-Based Objective Functions for Active Data Selection," *Neural Computation,* vol. 4, no. 4, pp. 590-604, 1992.
[26] D. Lewis and W. Gale, "A Sequential Algorithm for Training Text Classifiers," *Proc. Int'l SIG-IR Conf. Research and Development in Information Retrieval,* pp. 3-12, 1994.
[27] Z. Zheng and B. Padmanabhan, "On Active Learning for Data Acquisition," *Proc. IEEE Conf. Data Mining,* pp. 562-569, 2002.
[28] X. Zhu, X. Wu, and Y. Yang, "Error Detection and Impact-Sensitive Instance Ranking in Noisy Data Set," *Proc. 19th Nat'l Conf. Artificial Intelligence (AAAI),* 2004.
[29] X. Zhu and X. Wu, "Data Acquisition with Active and Impact-Sensitive Instance Selection," *Proc. IEEE Int'l Conf. Tools with Artificial Intelligence (ICTAI),* 2004.
[30] D. Lizotte, O. Madani, and R. Greiner, "Budgeted Learning of Naive-Bayes Classifiers," *Proc. Uncertainty in Artificial Intelligence,* 2003.
[31] P. Turney, "Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm," *J. Artificial Intelligence Research,* vol. 2, pp. 369-409, 1995.
[32] C. Blake and C. Merz, UCI Repository of Machine Learning Databases, 1998.
[33] M. Nunez, "The Use of Background Knowledge in Decision Tree Induction," *Machine Learning,* vol. 6, pp. 231-250, 1991.
[34] M. Tan, "Cost-Sensitive Learning of Classification Knowledge and Applications in Robotics," *Machine Learning,* vol. 13, 1993.
[35] P. Hoel, "Likelihood Ratio Tests," *Introduction to Mathematical Statistics,* third ed. New York: Wiley, 1962.
[36] C. Shannon and W. Warren, *The Mathematical Theory of Communication.* Univ. of Illinois Press, 1971.
[37] A. Freitas, "Understanding the Crucial Role of Attribute Interaction in Data Mining," *Artificial Intelligence Rev.,* vol. 16, no. 3, pp. 177-199, 2001.
[38] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Prentice Hall, 1998.

Xingquan Zhu received the PhD degree in computer science from Fudan University, Shanghai, China, in 2001. He spent four months with Microsoft Research Asia, Beijing, China, where he worked on content-based image retrieval with relevance feedback. From 2001 to 2002, he was a postdoctoral associate in the Department of Computer Science, Purdue University, West Lafayette, Indiana. He is currently a research assistant professor in the Department of Computer Science, University of Vermont, Burlington, Vermont. His research interests include data mining, machine learning, data quality, multimedia systems, and information retrieval. Since 2000, Dr. Zhu has published extensively, including more than 45 refereed papers in various journals and conference proceedings. He is a member of the IEEE.

Xindong Wu received the PhD degree in artificial intelligence from the University of Edinburgh, Britain. He is professor and chair of the Department of Computer Science at the University of Vermont. His research interests include data mining, knowledge-based systems, and Web information exploration. He has published extensively in these areas in various journals and conferences, including *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *IEEE Intelligent Systems*, *ACM Transactions on Information Systems*, IJCAI, AAAI, ICML, KDD, ICDM, and WWW. He has been an invited/keynote speaker at six international conferences, including the recent 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004) and the Joint 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2004) and 2004 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2004). Dr. Wu is the Executive Editor (January 1, 1999 - December 31, 2004) and an Honorary Editor-in-Chief (starting January 1, 2005) of *Knowledge and Information Systems* (a peer-reviewed archival journal published by Springer), the founder and current Steering Committee Chair of the IEEE International Conference on Data Mining (ICDM), a Series Editor of the Springer Book Series on Advanced Information and Knowledge Processing (AI&KP), and the Chair of the IEEE Computer Society Technical Committee on Intelligent Informatics (TCII). He served as an Associate Editor for the *IEEE Transactions on Knowledge and Data Engineering* between January 1, 2000 and December 31, 2003 before becoming editor-in-chief in 2005. He is the winner of the 2004 ACM SIGKDD Service Award. He is a senior member of the IEEE.