Int. J. Appl. Math. Comput. Sci., 2016, Vol. 26, No. 4, 871–884
DOI: 10.1515/amcs-2016-0062

LOOKAHEAD SELECTIVE SAMPLING FOR INCOMPLETE DATA

LOAI ABDALLAH [a,b,*], ILAN SHIMSHONI [c]

[a]Management Information Systems
The Max Stern Yezreel Valley College, Emek Yezraeel, 1930600, Israel
e-mail: aloai@is.haifa.ac.il

[b]Department of Mathematics and Computer Science
College of Sakhnin for Teacher Education, Sakhnin, 20173, Israel

[c]Department of Information Systems
University of Haifa, Haifa, 199, Israel
e-mail: ishimshoni@mis.haifa.ac.il

Missing values in data are common in real world applications. There are several methods that deal with this problem. In this paper we present lookahead selective sampling (LSS) algorithms for datasets with missing values. We developed two versions of selective sampling. The first one integrates a distance function that can measure the similarity between pairs of incomplete points within the framework of the LSS algorithm. The second algorithm uses ensemble clustering in order to represent the data in a cluster matrix without missing values and then run the LSS algorithm based on the ensemble clustering instance space (LSS-EC). To construct the cluster matrix, we use the k-means and mean shift clustering algorithms especially modified to deal with incomplete datasets. We tested our algorithms on six standard numerical datasets from different fields. On these datasets we simulated missing values and compared the performance of the LSS and LSS-EC algorithms for incomplete data to two other basic methods. Our experiments show that the suggested selective sampling algorithms outperform the other methods.

Keywords: selective sampling, missing values, ensemble clustering.

1. Introduction

Supervised learning algorithms require that a set of labeled examples be given to the algorithm in order to train a classifier. In many cases we want to construct a training dataset or add examples to the training dataset in order to improve the classifier's quality. In real environments, it is usually difficult to obtain a large set of labeled examples since each example must be labeled by a domain expert. It is therefore important to reduce the number of the training examples that were labeled by the expert. In other words, we should ask the expert to label the most informative unlabeled examples. To achieve this goal, we should provide the learning algorithm with some control over the inputs on which it trains. This paradigm is called *active learning*.

Selective sampling is one of the most common active learning approaches. It assumes that a set of unlabeled examples is available, and the learner selects an unlabeled example from the given set and asks the teacher to label it.

The problem with these approaches is that they cannot deal with datasets that contain missing values which are common in many real world datasets. Missing values can be caused by human errors, equipment failures, system generated errors, and so on. We were introduced to the problem of missing data when we received datasets from Applied Materials (AMAT), a company which develops inspection machines for the semiconductor industry.

To this end, in this research we developed two versions of a selective sampling algorithm that can deal with datasets with missing values. One version is for the original LSS described by Lindenbaum *et al*. (2004) while the other for LSS-EC described by Abdallah and Shimshoni (2013).

*Corresponding author

Studying the LSS algorithm we saw that to compute the expected utility all we need is a dissimilarity metric between the points and not the points themselves. As a result, in this research we decided to work with a variance of the *mean Euclidean distance* (MD_E) presented by Abdallah and Shimshoni (2014) to measure the dissimilarity between the incomplete points. The MD_E distance is not only efficient, but also takes into account the distribution of each attribute. This distance assumes that the missing values are randomly distributed across all the samples. In this case, samples with complete data are generated from the same distribution as those with incomplete data. But, in real world datasets, the missing values may depend on information from the known values of the data. As a result, in this research we generalize this distance function to deal with other types of missing values.

Abdallah and Shimshoni (2013) show that LSS using ensemble clustering (EC) performed better than LSS using the Euclidean distance. As a result, in this research we also decided to extend this algorithm (i.e., LSS-EC) for datasets with missing values. Ensemble clustering means running several clustering runs with different parameters or different clustering algorithms. Using ensemble clustering yields a new instance space where each data point is represented as the cluster's label at each clustering run. This new space has two important advantages. Firstly, there is a distance metric defined by Abdallah and Shimshoni (2013) that can reflect the similarity between the objects better than the Euclidean distance. Moreover, it is a new way to represent the incomplete data as complete categorical data using cluster labels. Ensemble clustering is general and any good clustering algorithm can be used. But since this research is for incomplete data, we use only clustering algorithms that are able to cluster incomplete datasets. In our experiments we decided to work with the mean shift clustering algorithm (Abdallah and Shimshoni, 2014) and the k-means clustering algorithm (Abdallah and Shimshoni, 2016), which were developed to deal with missing values.

To measure the performance of the suggested selective sampling algorithms we experimented on six standard numerical datasets from different fields from the Speech and Image Processing Unit (Clustering datasets, 2008). Our experiments show that the performance of the LSS using the MD_E distance (LSS-MD_E) and LSS-EC using the MD_E distance (LSS-EC-MD_E) algorithms with our distance function was superior to LSS using other methods.

The paper is organized as follows. Related work is discussed in Section 2. The LSS algorithm for incomplete datasets using the MD_E distance is described in Section 3. Section 4 presents lookahead selective sampling using ensemble clustering for incomplete datasets. Experimental results on numerical datasets are given in Section 5. Finally, our conclusions are presented in Section 6.

2. Related work

Previous work on the problem of selecting a sample of relevant instances from a set of unlabeled data falls under the paradigm of active learning and, more specifically, *selective sampling*, which is more common in practice. Here it is assumed that a set of unlabeled examples is available. In this approach the learner selects an unlabeled example from the given set and asks the teacher to label it.

Much work has been done in selective sampling of examples related mainly to training classifiers. The most simple sampling technique is random sampling, where we select a group of instances from the instance space. Another option is to select the point with the largest uncertainty (Lewis and Gale, 1994).

Lindenbaum *et al*. (2004) claimed that the problem of selective sampling is similar to that of cost-sensitive learning (Tan and Schlimmer, 1990; Turney, 1995). They proposed the *lookahead algorithm for selective sampling* (LSS) for the nearest neighbor classifier. The main goal of their algorithm is to develop a selective sampling methodology for nearest-neighbor (NN) classification learning algorithms. Later, Abdallah and Shimshoni (2013) developed a selective sampling algorithm based on ensemble clustering.

Dasgupta and Hsu (2008) use hierarchical clustering. Their method exploits the cluster structure (if there is any) in the unlabeled data. Their algorithm assumes that querying the label of only one of the data points in a cluster is sufficient to determine the label of the other data points in that cluster.

Similarly, Xu *et al*. (2007) use clustering to construct sets of queries for batch-mode active learning with SVMs. Specifically, they query the centroids of clusters of instances that lie closest to the decision boundary.

Hospedales *et al*. (2013) developed an active learning algorithm to optimize both rare class discovery and classification simultaneously. In their paper they addressed these issues with two contributions: a unified active learning model to jointly discover new categories and learn to classify them by adapting query criteria online, and a classifier combination algorithm that switches generative and discriminative classifiers as learning progresses.

Lughofer (2012) proposed a novel active learning strategy for data-driven classifiers, which is essential for reducing the annotation and supervision effort of operators in off-line and on-line classification systems, as operators only have to label an exquisite subset of off-line training data. Li *et al*. (2012) proposed a joint active learning approach which combines a novel generative

query strategy and the existing discriminative one, which adaptively fits the distribution difference and shows higher robustness than the ones using a single strategy.

Zhang *et al*. (2014) combined the benefits of both co-training and active learning to propose a new semi-supervised learning algorithm. The algorithm applies co-training to select the most reliable instances according to the two criterions of high confidence and nearest neighbor for boosting the classifier, as well as to exploit the most informative instances with human annotation to improve the classification performance.

Dekel *et al*. (2012) introduced an online active learning algorithm, designed to incrementally make binary predictions on a sequence of adversarially generated instances. However, they can also convert their algorithm into an efficient statistical active learning one working under standard statistical assumptions, which receives a sample of instances from some unknown distribution, queries the teacher for a subset of the labels, and outputs a hypothesis with a small error rate. This work did not take into account the difference between the teachers. For example, one could imagine a setting where the cost of each label depends on each teacher's confidence in his/her own answer. In consequence, it would be interesting to extend our work to a setting where different teachers charge different rates. For example, one could imagine a setting where the cost of each label depends on each teacher's confidence in his/her own answer.

The problem of these algorithms is that they cannot deal with incomplete datasets. This is an important problem, since missing values are common in real world datasets. Based on the work of Donders *et al*. (2006), Ibrahim *et al*. (2005), Little (1988), and Little and Rubin (2014), there are three basic types of missing data:

1. Missing completely at random: data are said to be MCAR if the failure to observe a value is not related to any other sample.

2. Missing at random: data are said to be MAR if the probability that a value is missing does not depend on the other missing values. Thus the conditional probability of missingness may depend on any known values.

3. Not missing at random: data are said to be NMAR if the probability that a known value is missing depends on the value that would have been observed.

Several methods have been proposed to deal with missing data. These methods can be classified into three basic categories:

(a) The case deletion method, which assumes that the missing values are missing completely at random (MCAR). It therefore ignores all the instances with missing values and performs the analysis on the rest (Zhang *et al*., 2005).

(b) Missing data imputation, which replaces each missing one with a known value according to the dataset distribution.

A common method that imputes missing data is the most common attribute (MCA) value method. The value of the attribute that occurs most often is selected to be the value for all the unknown values of the attribute (Grzymala-Busse and Hu, 2001). The mean imputation (MI) method replaces a data point with missing values with the mean of all the instances in the data. A variant of this method is to replace the missing data for a given attribute with the mean of all known values of that attribute (MA) (i.e., the mean of each attribute) in the coordinate where the instance with missing data belongs (Magnani, 2004). All these methods assume MCAR since each one is based on the distribution of the whole data and does not take into account the correlations between the observed and unobserved values.

It is important to note that by using these methods the LSS algorithm can run like on the complete dataset (each missing item replaced with a known value). But, as we show in this paper, they perform poorly and our proposed methods yield better results.

(c) Rough set theory, which deals with four kinds of missing attributes values: lost values (the values that were recorded but currently are unavailable), "do not care" conditions (the original values were irrelevant), restricted "do not care" conditions (similar to ordinary "do not care" conditions but interpreted differently), and attribute-concept values (these missing attribute values may be replaced by any attribute value limited to the same concept) (Stefanowski and Tsoukias, 2001; Grzymala-Busse, 2006; Bai *et al*., 2015). Clark *et al*. (2013) use rough sets theory to deal with consistency of incomplete data sets (a data set is defined as consistent when any pair of samples with the same attribute values belongs to the same class); they discuss two types of missing attribute values: lost values and "do not care" conditions. In their work, two possible types of consistencies are defined using probabilistic approximations. The first type occurs if there exists some probability for which the corresponding approximation of the type singleton, subset or concept is equal to the concept. In the second type of consistency, for any probability the corresponding approximations are all equal to the concept. Nowicki *et al*. (2016) proposed an application of rough sets in the k-nearest-neighbor algorithm for classification of incomplete samples.

Another work of Nowicki (2010) presents a new approach to fuzzy classification in the case of missing data. Rough-fuzzy sets are incorporated into logical type neuro-fuzzy structures and a rough-neuro-fuzzy classifier is derived. The main benefit of the proposed modification is protection against mistakes in the case of missing input data. The complexity of the proposed solution is similar to that of other fuzzy and neuro-fuzzy systems.

Abdallah and Shimshoni (2014) developed a new method to compute the distance function between incomplete points. Their distance is not only efficient, but also takes into account the distribution of each attribute. In the computation procedure they take into account all the possible values with their probabilities, which are computed according to the attribute's distribution. This is in contrast to MCA and the MA methods, which replace each missing value only with the mode or the mean of each attribute.

The main limitation of this distance is that it assumes MCAR (the probabilities were computed according to the distributions of the whole datasets without taking into account any correlations between the missing values and the other known values). But in real world datasets, the MCAR assumption is too stringent for some situations. As a result, in this research we generalize this distance to deal with other types of missing data.

3. Lookahead algorithm for selective sampling for incomplete datasets

In this research we develop two versions of the selective sampling algorithm that deal with incomplete datasets. The first one is based on *selective sampling for the nearest neighbor classifier* (LSS) (Lindenbaum *et al.*, 2004), and the second one is based on *selective sampling using ensemble the clustering based distance metric* (LSS-EC) (Abdallah and Shimshoni, 2013). In this section we will describe the LSS algorithm for incomplete datasets. In order to run this algorithm on an incomplete dataset, we need a distance function that can measure the distance between incomplete points. In this research we decide to work with the *mean Euclidean distance* (MD_E) presented by Abdallah and Shimshoni (2014) to measure the dissimilarity between incomplete points. We will therefore first review the LSS algorithm. Then we will give a short review of the MD_E distance. Finally, we will describe how we integrated this distance within the framework of LSS.

3.1. Original lookahead algorithm for selective sampling.

In this section we will describe the original LSS for complete datasets developed by Lindenbaum *et al.* (2004). *Selective sampling* is a common active learning approach. It assumes that a set of unlabeled examples is available, and the learner selects an unlabeled example from the given set and asks the teacher to label it.

An *active-learner* consists of a classifier learning algorithm L and a selective sampling algorithm S_L. The selective sampling algorithm determines which unlabeled instance in X should be labeled by the teacher f, which is a mapping $f : \chi \to \mathcal{W}$, where χ is the instance space and $\mathcal{W}$ describes the classes.

The active learner first applies S_L to choose one unlabeled instance x from X. The label w of x is then revealed and the pair (x, w) is added to the training set D and x is removed from X. Then the learner applies L to induce a new classifier. This sequence repeats until some stopping criterion is satisfied.

The *lookahead* algorithm for selective sampling considers all the unlabeled examples and selects the example that yields the best expected classifier. Let $U_L(x, D)$ be a *utility function* that estimates the merit of adding an unlabeled instance x to the set D as a training example for the learning algorithm L. Let $P(f(x) = w|D)$ denote the conditional class probabilities of x for a given labeled set D. For each unlabeled example, its expected utility is measured using the utility function on the training set and employing expected probabilities for the possible classes of the unlabeled example. Then the lookahead algorithm for selective sampling with respect to the learning algorithm L selects the example that leads to the learning example with the highest expected utility. This is described in Algorithm 1.

In order to be able to use this general algorithm for a specific learner, $A_L(D)$ and $P(f(x) = w|D)$ have to be defined. $A_L(D)$ denotes the expected accuracy of classifier $h = L(D)$ as

$$A_L(D) = \frac{1}{|X|} \sum_{x \in X} P(f(x) = h(x)|D), \qquad (1)$$

is produced by a learning algorithm L on labeled data D.

The last piece of the puzzle is to assign conditional class probabilities to the nearest neighbor classifier. To this end, the random field model is used to estimate the probabilities for the 2-NN classifier.

In order to calculate $P(f(x) = 1|D)$, first the two nearest neighbors y, z from the labeled data D must be found. Then the probability is computed as

$$P(f(x) = 1|y, z) = \frac{1}{2} + \frac{l(y)\gamma(d(x, y)) + l(z)\gamma(d(x, z))}{\frac{1}{2} + 2l(y)l(z)\gamma(d(y, z))}, \qquad (2)$$

where $l(x)$ is the label of x and $d(x, y)$ is the distance between x and y, and

Algorithm 1. LSS (lookahead selective sampling) (X, D).

1. If D is empty, return a random point from X.
2. Otherwise, set $U_{\max} \leftarrow 0$.
3. For each $x \in X$ do:
   (a) $D' \leftarrow D \cup \{\langle x, -1\rangle\}$.
   (b) Compute class probabilities for all points in X based on data D'.
   (c) Compute utility by approximating the accuracy $A_L(D')$ of the classifier based on data D', $U_1 \leftarrow A_L(D')$.
   (d) Repeat steps (a)–(c) for $D' \leftarrow D \cup \{\langle x, 1\rangle\}$ and get $U_2(x)$.
   (e) Compute class probabilities for x based on data D.
   (f) $U(x) \leftarrow P(f(x) = -1|D) \cdot U_1(x) + P(f(x) = 1|D) \cdot U_2(x)$.
   (g) If $U_{\max} < U(x)$ then $U_{\max} \leftarrow U(x)$ and $x_{best} \leftarrow x$.
4. Return x_{best}.

$$\gamma(d) = 0.25e^{-d/\sigma}, \quad \sigma = \frac{1}{\mathfrak{D}} \frac{1}{|N|^2} \sum_{p \in X} \sum_{q \in X} d(p, q),$$

where $\mathfrak{D}$ is a scaling parameter.

In order to run the LSS algorithm on a incomplete dataset, we need to compute the probabilities using Eqn. (2). To do this, we should take into account the completeness of the points x, y and z. In order to compute the probability, we need the labels and the distances between each point to other points. To compute the distance, we decided to use the MD_E distance as described at the beginning of this section. Thus, in the next section we will give a short review of this distance, developed by Abdallah and Shimshoni (2014).

3.2. Mean Euclidean distance on incomplete datasets.

In this section we describe how to measure the distance between pairs of points when they may contain missing values.

Let $A \subseteq \mathbb{R}^K$ be a set of points. For the i-th attribute A^i, the conditional probability for A_i will be computed according to the known values for this attribute from A (i.e., $P(A^i) \sim \chi^i$), where χ^i is the distribution of the i-th coordinate.

Given two sample points X and Y from A, the goal is to compute the distance between them. Let x^i and y^i be the i-th coordinate values from points X, Y, respectively. There are three possible cases for the values of x^i and y^i:

1. Two values are known: when the values of x^i and y^i are given, the distance between them will be defined as the Euclidean distance,
$$D_E(x^i, y^i) = (x^i - y^i)^2. \tag{3}$$
2. One value is missing: suppose that x^i is missing and the value y^i is given. Since the value of x^i is unknown, we cannot compute its Euclidean distance. Instead, we model the distance as a random selection of a point from the distribution of its attribute χ^i and compute its distance. The expectation of this computation is our distance.

   As a result, we approximate the mean Euclidean distance (MD_E) between y^i and the missing value m^i as
$$\begin{aligned} MD_E(m^i, y^i) &= E[(x - y^i)^2] \\ &= \int p(x)(x - y^i)^2 \, \mathrm{d}x \\ &= (y^i - \mu^i)^2 + (\sigma^i)^2. \end{aligned} \tag{4}$$
   This metric measures the distance between y^i and each suggested value of x^i, and takes into account the probability $p(x)$ for this value according to the evaluated probability distribution. It is important to note that in this computation the probability was computed according to the whole dataset. The authors did not take into account the possible correlations between the missing values and the other known values. This means that they assumed the MCAR (missing completely at random) missing data type. The resulting mean Euclidian distance will be
$$MD_E(m^i, y^i) = (y^i - \mu^i)^2 + (\sigma^i)^2, \tag{5}$$
   where μ^i and $(\sigma^i)^2$ are the $mean$ and the $variance$ for all the known values of the attribute. The distance computed according to the last equation has several important properties:
   - it is identical to the Euclidean distance when the dataset is complete;
   - it can be applied to any distribution and used in any algorithm that employs a distance function;
   - it is simple to implement;
   - it is very efficient because, to compute the MD_E between two values when one of them is missing, we need only to compute in advance the two statistics (i.e., μ and σ) for each attribute. This means that, after we compute them, the runtime to measure the distance is $O(1)$;

- when the variance is small, the real value of the missing value is close to the mean of the attribute and our distance will converge to the Euclidian one;
- when the variance is large, the uncertainty is high, and as a result the distance should be large;
- it is basically the sum of the bias term $(y^i - \mu^i)^2$ and the $variance$ $(\sigma^i)^2$, yielding the mean squared error (MSE).

3. The two values are missing: in this case, in order to estimate the mean Euclidian distance, we have to randomly select values for both x^i and y^i. Both of these values are selected from distribution χ^i.

   We compute the expectation of the Euclidean distance between each selected value as we did for the one missing value. As a result, the distance is

$$MD_E(x_i, y_i) = \iint p(x)p(y)(x-y)^2 \,\mathrm{d}x\,\mathrm{d}y = (E[x] - E[y])^2 + \sigma_x^2 + \sigma_y^2.$$

   As x and y belong to the same attribute, $E[x] = E[y] := \mu^i$ and $\sigma_x = \sigma_y := \sigma^i$. Thus

$$MD_E(x^i, y^i) = 2(\sigma^i)^2. \tag{6}$$

Based on this distance function, a formula for the mean of a set with missing values derived according to this distance is equal to the mean of the known values of each coordinate.

Studying the equation described above, we conclude that the main limitation of this distance is its assumption that the missing data is MCAR. However, many real world datasets are not MCAR. This is because the probabilities that were computed for each suggested value of the missing value do not take into account the type of the missing values. Accordingly, if the missing values are MAR (missing at random), then the probability $p(x)$ depends on the other observed values, and then the distance will be computed as:

$$\begin{aligned} MD_E(m^i, y^i) &= \int p(x|x_{obs})(x-y^i)^2 dx \\ &= (y^i - \mu^i_{x|x_{obs}})^2 + (\sigma^i_{x|x_{obs}})^2, \end{aligned}$$

where x_{obs} denotes the observed attributes of point X, and $\mu^i_{x|x_{obs}}$ and $(\sigma^i_{x|x_{obs}})^2$ are the conditional $mean$ and $variance$, respectively.

On the other hand, if the missing values are of type NMAR (not missing at random), this means that whether a known value is missing depends on the value that would have been observed, and then the probability $p(x)$ that was used in Eqn. (4) will be computed according to this information and so the distance will be

$$\begin{aligned} MD_E(m^i, y^i) &= \int p(x|m^i)(x-y^i)^2 \mathrm{d}x \\ &= (y^i - \mu^i_{x|m^i})^2 + (\sigma^i_{x|m^i})^2, \end{aligned}$$

where $p(x|m^i)$ is the distribution of x when x is missing.

3.3. LSS using the MD_E distance for incomplete datasets. Now we will show how to integrate the distance function described above within the framework of the LSS algorithm in order to run this algorithm on incomplete datasets. This distance function, after our generalization, can deal with the different types of missing values. However, for simplicity, in this section we will describe only the MCAR missing value type.

The LSS algorithm (i.e., Algorithm 1) consists of several steps. Formally, we need to establish the utility function $U_L(x, D)$ that estimates the merit of adding an unlabeled instance x to the training data D, taking into account that x or D may contain missing values. However, in order to deal with missing values, all we need is to modify Eqn. (2) to compute the probabilities when some of the points are incomplete. Thus, now we will describe how to design probability computation in order to deal with missing values. Since in this research we decided to work with 2-NN probability, in probability computation we have three points, x is the point for which we want to compute its probability, and y and z are the two nearest samples to x. Because all the points come from the data, each point may contain missing values. To this end, in our computation we take into account all the completeness possibilities for each point as described below:

(a) x is complete, which implies there are three possible cases for y and z:

1. Both of them are complete—then we use the Euclidean distance.

2. One of them contains missing values: let y be the incomplete point and let the coordinate j be the missing value; then

$$d^2(x, y) = \sum_{k \neq j} (x^k - y^k)^2 + ((x^j - \mu^j)^2 + (\sigma^j)^2),$$

$$d^2(x, z) = \sum_{k=1}^{n} (x^k - y^k)^2,$$

$$d^2(y, z) = \sum_{k \neq j} (z^k - y^k)^2 + ((z^k - \mu^j)^2 + (\sigma^j)^2).$$

3. Both of them contain missing values: y and z are incomplete; let j_1, j_2 be the missing values in y, z, respectively; then

$$d^2(x,y) = \sum_{k \neq j_1} (x^k - y^k)^2 + ((x^{j_1} - \mu^{j_1})^2 + (\sigma^{j_1})^2),$$

$$d^2(x,z) = \sum_{k \neq j_2} (x^k - z^k)^2 + ((x^{j_2} - \mu^{j_2})^2 + (\sigma^{j_2})^2),$$

and, if $j_1 \neq j_2$, then

$$d^2(y,z) = \sum_{k \neq j_1, j_2} (y^k - z^k)^2 + ((z^{j_1} - \mu^{j_1})^2 + (\sigma^{j_1})^2) + ((y^{j_2} - \mu^{j_2})^2 + (\sigma^{j_2})^2),$$

otherwise

$$d^2(y,z) = \sum_{k \neq j_1} (y^k - z^k)^2 + (2(\sigma^{j_1})^2).$$

(b) x contains missing values—let the coordinate l be the missing value. Below we will describe the designed equations for distance computations:

1. Both of them are complete, which yields

$$d^2(x,y) = \sum_{k \neq l} (x^k - y^k)^2 + ((y^l - \mu^l)^2 + (\sigma^l)^2),$$

$$d^2(x,z) = \sum_{k \neq l} (x^k - z^k)^2 + ((z^l - \mu^l)^2 + (\sigma^l)^2),$$

$$d^2(y,z) = \sum_{k=1}^{n} (y^k - z^k)^2.$$

2. One of them contains missing values: let y be the incomplete point and let the coordinate j be the missing value. If $j \neq l$,

$$d^2(x,y) = \sum_{k \neq j, l} (x^k - y^k)^2 + ((y^l - \mu^l)^2 + (\sigma^l)^2) + ((x^j - \mu^j)^2 + (\sigma^j)^2),$$

and, if $l = j$, then

$$d^2(x,y) = \sum_{k \neq l} (x^k - y^k)^2 + 2(\sigma^l)^2,$$

$$d^2(x,z) = \sum_{k \neq l} (x^k - z^k)^2 + ((z^l - \mu^l)^2 + (\sigma^l)^2),$$

$$d^2(y,z) = \sum_{k \neq j} (z^k - y^k)^2 + ((z^j - \mu^j)^2 + (\sigma^j)^2).$$

3. Both of them contain missing values: in a similar way, we compute the distances between the points when y and z are incomplete.

According to these computations, we can now compute the probabilities for each point in the data. As a result, we can run the LSS algorithm on an incomplete dataset using the MD_E distance function.

4. Selective sampling using ensemble clustering for incomplete datasets

We will now describe LSS for incomplete data based on the ensemble clustering. Accordingly, we firstly describe distance metric learning using ensemble clustering. Then we will describe how we run the ensemble clustering on datasets with missing values. The third part of this section describes the changes we made to the original LSS formulas to cause the algorithm to work with the new distance metric and the new instance space.

4.1. Distance metric learning using ensemble clustering. Let A be a set of instances, where each $x_i \in A$ is a vector in some space χ. Instances are assumed to be i.i.d. distributed according to some unknown fixed distribution ρ.

In our approach the clustering results are stored in a matrix denoted as the *cluster matrix* $C \in \mathbb{R}^{N \times K}$, where $N = |A|$ and K is the number of times the clustering algorithms were run. The i-th row consists of the cluster identities of the i-th point in the different runs. This results in a new instance space, $\chi_{cl} = \mathbb{Z}^K$, which contains the rows of the *cluster matrix*.

It is important to note that the new instance space χ_{cl} is a new way to represent the dataset. With this instance space we overcome the problem of missing data and move from the continuous space with missing values to the categorical space without missing values.

The new distance between points from this space should be defined in order to reflect our intuitive notion of proximity among the corresponding points.

Given two points $x, y \in \chi_{cl}$, we define a new distance function d_{cl} as

$$d_{cl}(x,y) = \sum_{i=1}^{K} dis(x_i, y_i), \tag{7}$$

where

$$dis(x_i, y_i) = \begin{cases} 1, & x_i \neq y_i, \\ 0, & x_i = y_i \end{cases}$$

is the metric of a single feature. This metric is known as the *Hamming distance*. The idea of measuring similarity between objects according to their clustering labels was introduced by Strehl and Ghosh (2002). Moreover, points

which were always clustered together in the same clusters are defined as members of an equivalence class.

The main problem with a clustering based approach is that there is no known method for choosing the best clustering. It is unknown how many clusters there should be, and so are their shapes, which clustering algorithm is best, and which parameter values should be used. We therefore decided to run different clustering algorithms several times with different parameter values. In general, each clustering algorithm that maps the dataset structure and yields close to pure equivalence classes will work. In addition, the clustering matrix can contain clustering results from different clustering algorithms.

4.2. Ensemble clustering for incomplete datasets. As described in the previous section, our algorithm is general and any good clustering algorithm could be used. Therefore, in our case the datasets contain missing values. In consequence, we need to run clustering algorithms that can deal with datasets of this type. As a result, we decided to use the k-means clustering algorithm (Abdallah and Shimshoni, 2016) and the mean shift clustering algorithm (Abdallah and Shimshoni, 2014), designed to cluster incomplete datasets in order to build the cluster matrix.

For completeness, we will now give a short overview of the k-means and mean shift clustering algorithms we use to cluster incomplete datasets. Here we only review some of the results described earlier in detail (Abedallah and Shimshoni, 2016; 2014).

- k-means (MacQueen, 1967) is the most popular and the simplest partitional clustering algorithm. It has two basic steps, performed at each iteration: (i) it associates each point with its closest centroid, and (ii) it computes the new centroids. They use the MD_E distance function to associate each point to the closest centers, and to compute the mean three directions are suggested. In the first one, incomplete points are dealt with as one point and the centroid is computed according to the developed formula based on the known values at each coordinate. The other two directions assume that each incomplete point represents a set of complete points according to the data distribution, so they replace each incomplete point with a set of points and then compute the $mean$ according to the new dataset. It is important to note that even though they replace each incomplete point with a large number of points, they use the histograms of the data distribution in order to make the suggested algorithm more efficient. As a result, the runtime complexity of the suggested k-means algorithms is the same as the standard k-means over complete datasets.

- Mean shift is a non-parametric clustering algorithm. In addition, it is an iterative technique, but instead of the mean, it estimates the modes of the multivariate distribution underlying the feature space. The number of clusters is obtained automatically by finding the centers of the densest regions in the space (the modes).

  Assume that each data point $x_i \in \mathbb{R}^d, i = 1, \ldots, n$, is associated with a bandwidth value $h > 0$. The *sample point* density estimator at point x is

$$\hat{f}_{h,k}(x) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k\left(\left\|\frac{x - x_i}{h}\right\|^2\right), \qquad (8)$$

  where the function $k(x), 0 \leq x \leq 1$, is called the *profile* of the kernel, and the normalization constant $c_{k,d}$ assures that the kernel integrates to one.

  Using the MD_E distance, the density estimator in (8) is written as

$$\begin{aligned}\hat{f}_{h,k}(x) &= \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \\ &= \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k\left(\frac{1}{h^2} \sum_{j=1}^{d} MD_E(x^j, x_i^j)^2\right).\end{aligned}$$

  For each data point, a gradient ascent process is performed on the local estimated density until convergence. The convergence points represent the modes of the density function. All points associated with the same convergence point belong to the same cluster.

Some clustering algorithms work with continuous parameters, such as the mean shift algorithm described above. In these cases the differences between two consecutive iterations might be small. We therefore represent similar clusterings by a single column, weighted by the number of clusterings it represents.

It is important to note that using EC has several important features:

- a new data structure, where the points are now represented in a categorical space by the labels of the clusters at each iteration;
- it is a new way to overcome the missing values without replacing it with any value;
- defining a similarity measurement over incomplete datasets.

4.3. Selective sampling using ensemble clustering for incomplete datasets. We will now describe the changes we made to the original LSS formulas to cause

the algorithm to work with the new distance metric and the new instance space.

We are looking for x that maximizes the utility function $U_L(x, D_{cl})$ described in the previous sections to be added to the training data. It is important to note that in this situation each point x is an equivalence class, so we should take into account the cardinality of each point x, which is the size of the equivalence class of x.

In each iteration of the LSS-EC algorithm, one of the equivalence classes is chosen to be added to the training dataset. Its representative point x_0 is labeled by the expert and added to D_{cl}. This is in contrast to the original LSS, where a single point is chosen. In order to choose the equivalence class, the expected accuracy will be

$$A_L(D'_{cl}) = \frac{1}{|X|}\left[\sum_{x\in D_{cl}} P_{\max}\left(x|D'_{cl}\right)\cdot card\left(x\right) + P_{\max}\left(x_0|D'_{cl}\right)\cdot card\left(x_0\right) + \sum_{x\in X^{eq}_{cl}\setminus D'_{cl}} P_{\max}\left(x|D'_{cl}\right)\cdot card\left(x\right)\right],$$

where X^{eq}_{cl} is a set of equivalence points, $D'_{cl} = D_{cl} \cup \{x_0\}$, and $card(x)$ is the cardinality of x, which is the size of the equivalence class of x. The probability $P_{\max}(x|D'_{cl})$ of each point is multiplied by its cardinality because in the original space X each point represents $|[x]_E|$ points which we assume have the same label. $P_{\max}\left(x \in D_{cl}|D'_{cl}\right) = 1$ because D_{cl} is labeled. As a result, the expected accuracy will be

$$A_L(D'_{cl}) = \frac{1}{|X|}\left[\sum_{x\in D_{cl}} card\left(x\right) + card\left(x_0\right) + \sum_{x\in X^{eq}_{cl}\setminus D'_{cl}} P_{\max}\left(x|D'_{cl}\right)\cdot card\left(x\right)\right].$$

The contribution of the point x_0 to the utility function is

$$\frac{1}{|X|}\Big[(1 - P_{\max}(x_0|D_{cl}))\cdot card(x_0) + \sum_{x\in X^{eq}_{cl}\setminus D'_{cl}} \left(P_{\max}\left(x|D'_{cl}\right) - P_{\max}\left(x|D_{cl}\right)\right)\cdot card\left(x\right)\Big]. \quad (9)$$

Studying the two components of (9), we can see that the first one, which is termed the uncertainty component, is the product of the cardinality of the set of points equivalent to x_0 with the reduction in uncertainty obtained since x_0 has been chosen. $P_{\max}\left(x_0|D_{cl}\right)$ is the probability of x_0 to be classified correctly in the past, and now since x_0 is chosen its probability will be 1. The second component, which is termed the classifier component, estimates the added contribution to the utility function, which will be obtained by reducing the uncertainty of the neighbors of x_0 to less than what it was for set D_{cl}. This term measures how the classifier improves when x_0 is added to the labeled set.

We ran the lookahead selective sampling algorithm using the d_{cl} metric and the new accuracy measure to choose the most informative points to be given to the expert. The rest of the points were labeled by the classifier h using the d_{cl} metric.

In conclusion, to run the LSS-EC over an incomplete dataset, all we need is to construct the cluster matrix, which we compute using the k-means and mean shift clustering algorithms, which deal with missing values described in Section 4.2.

5. Experiments on numerical datasets

In order to evaluate the performance of the suggested LSS-MD_E and LSS-EC-MD_E algorithms over datasets with missing values we compare the performance of the original LSS algorithm on complete data (i.e., without missing values) to its performance on data with missing values, using our distance measure (MD_E), and then again using LSS-(MCA, MA), where each missing value in each attribute is replaced using the MCA or MA methods (described in Section 2), respectively, and then the standard LSS is run.

We ran our experiments on six standard numerical datasets from the Speech and Image Processing Unit (Clustering datasets, 2008) from different fields: the Flame dataset, the Jain dataset, the Path based dataset, the Spiral dataset, the Compound dataset, and the Aggregation dataset. The dataset characteristics are shown in Table 1.

Table 1. Speech and Image Processing Unit dataset properties.

Dataset	Dataset size	Clusters
Flame	240×2	2
Jain	373×2	2
Path based	300×2	3
Spiral	312×2	3
Compound	399×2	6
Aggregation	788×2	7

All these datasets were labeled, but this knowledge was used only to evaluate the quality of the resulting classifier. In all the experiments the algorithm assumes that these datasets are unlabeled. Originally, these datasets were complete, and in our experiments a subset of these points were selected randomly to be incomplete.

In the first stage of the algorithms, 30% of the data were selected to be incomplete points, and then a

training set of size 4 was randomly drawn and labeled. The algorithms were then asked to select 20 additional points. During each iteration the active learner selects a sample point to be labeled and adds it to the training set. After each iteration the accuracy was evaluated by the ability of the classifier to label the rest of the unlabeled points. The results were averaged over 10 different runs of the algorithms on each dataset. For each dataset we constructed curves to evaluate how well the algorithms select the points.

To build the cluster matrix for the LSS-EC-MD_E algorithm, we use the mean shift or k-means clustering algorithms that were designed to deal with missing values, as discussed in Section 4.2. We arbitrarily chose to start with the k-means algorithm for all the numerical datasets with $k = 2, \ldots, 15$. Since the k-means algorithm did not yield pure equivalence classes for the spiral dataset while the mean shift clustering algorithm did, we used the mean shift algorithm for this dataset. In our experiments we chose 11 different values of h with fixed intervals from 0.5 to 5. It is important to note that the values of h are not critical for algorithm performance, and any clustering results that yield pure equivalence classes can work. The results are stored in the cluster matrix C. The new space is usually smaller than the original one without the equivalence classes.

As can be seen in Fig. 1, for the Flame, Spiral, Path based, Compound, and Aggregation datasets, the curves show that our suggested algorithms outperform their competitors over all these datasets. According to Fig. 1(e), for example, we see that the accuracy of LSS using the MD_E distance is 85% and the accuracy of LSS-EC-$Missing$ is 75% after selecting 6 points, while the accuracy of LSS using the other methods is only 60-70%.

An interesting result is that for the Jain dataset (as shown in Fig. 1(b)), where the performance of LSS-EC-MD_E on incomplete datasets outperforms the standard LSS on a complete dataset. This means that, although there are missing values, the performance of LSS-EC over the dataset with missing values is better than that of LSS using the Euclidean distance over the complete dataset. This is because for this dataset the distance based on ensemble clustering reflects the actual similarity between the objects better than the Euclidian distance even when there are missing values in the dataset. This means that representing the incomplete dataset according to the ensemble clustering space that is described in this research may be better than using the Euclidean distance on the actual values of the missing values in the dataset. It is important to note that, in general, the performance of the standard LSS on the complete dataset outperforms the other algorithms when the dataset is incomplete, and only in the case when the Euclidean distance does not reflect the actual similarity between the data objects LSS-EC-MD_E can achieve better results on incomplete datasets, as we can see in the other datasets in Fig. 1.

According to Fig. 1, in some cases the next step gives worse results than the previous one. This may be caused by two reasons:

(i) in LSS–EC we worked with equivalence classes and we assume that each equivalence class is pure, which is not correct in all the classes, and in some cases the class will be labeled by the minority label and will cause mistakes.

(ii) The performance of the algorithm was computed using $1NN$ over incomplete datasets because our algorithm selects points without filling the missing values; then the classifier will be asked to classify all the points according to the training set, which may include incomplete points.

Looking at the curves in Fig. 1 in general, we can see that, when we use LSS, the results are better than when we select the points randomly. That means that using selective sampling methods reduces the number of selected points that the domain expert has to classify and brings us more informative samples as well as better classification results.

Another experiment was conducted where the percentage of missing values changes. In this experiment we asked the algorithm to select 20 points from each dataset, where 5–50% of the data were selected to be incomplete.

As can be seen in Fig. 2, for the Spiral, Path based, Compound, and Aggregation datasets, the curves show that our two suggested algorithms (LSS-MD_E and LSS-EC-MD_E) outperformed the other methods for all missing value percentages, while for the Jain dataset the performance of the LSS-EC-MD_E algorithm, where 5–30% of the data were incomplete, was superior and outperformed the original LSS over the complete data (i.e., the original data without missing values). This is because running LSS over the new instance space based on ensemble clustering results over incomplete datasets reflects the actual similarity better than the Euclidean distance even though the dataset is complete. Moreover, we can conclude that, as the percentage of the missing values further increases, the performance of the algorithm degrades gracefully.

6. Conclusions and future work

Missing attribute values are very common in real-world datasets. Several methods have been proposed to measure the similarity between objects with missing values. To this end, in this research we developed two versions of the selective sampling algorithm that can deal with datasets

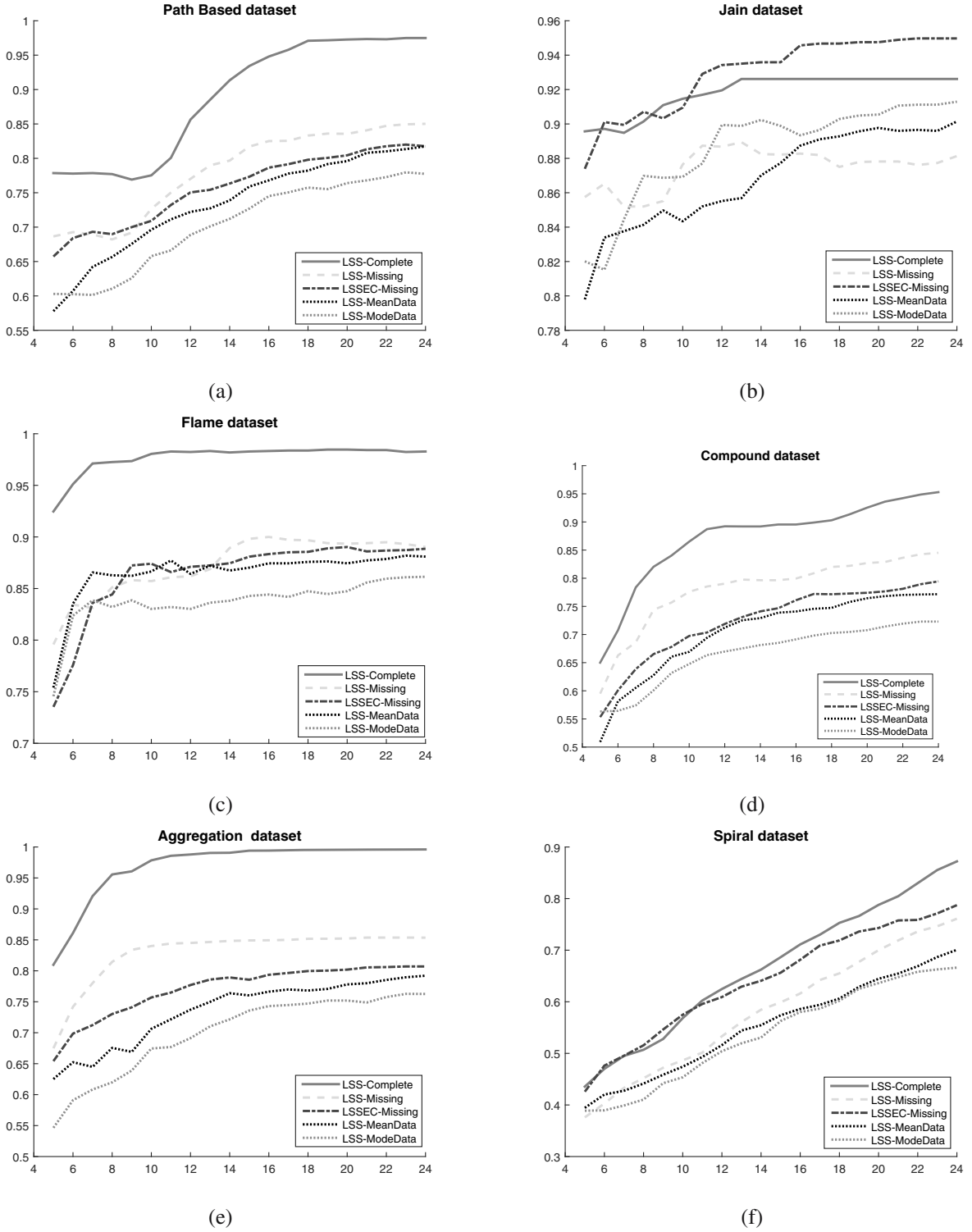

Fig. 1. Results of the original LSS on complete datasets and the suggested algorithms on incomplete datasets when 20 points were selected. The x axis denotes the number of selected points, and the y axis the accuracy of the kNN classifier. The solid curve describes the original LSS results on a complete dataset, the dashed and dash-dotted curves describe the LSS-MD_E and LSS-EC-MD_E results on incomplete datasets, respectively, while the black and grey dotted curves describe LSS using the MA and MCA methods, respectively. (Best viewed in color.)

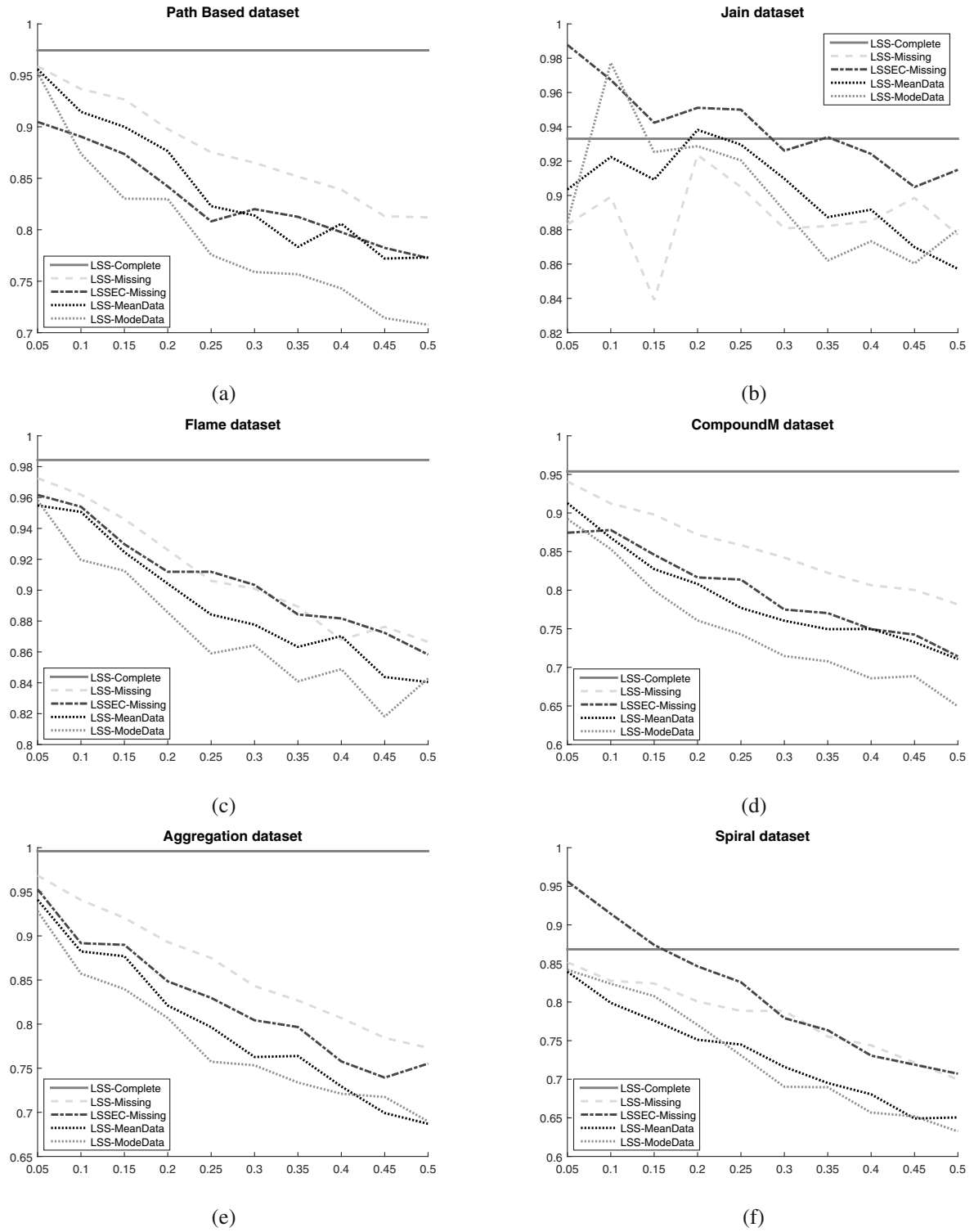

Fig. 2. Results of the original LSS on complete datasets and the suggested algorithms on incomplete datasets when 20 points were selected, where the percentage of the missing values changes. The x axis denotes the percentage of the missing values in the data, and the y axis denotes the accuracy of the kNN classifier. The solid curve describes the original LSS results on a complete dataset, the dashed and dash-dotted curves describe the LSS-MD_E and LSS-EC-MD_E results on incomplete datasets, respectively, while the black and grey dotted curves describe LSS using the MA and MCA methods, respectively. (Best viewed in color.)

with missing values. One version is for the original LSS described by Lindenbaum *et al*. (2004) using the MD_E distance that was presented by Abdallah and Shimshoni (2014), and the other is for LSS-EC described by Abdallah and Shimshoni (2013) using clustering algorithms that were designed to deal with incomplete datasets presented by Abdallah and Shimshoni (2014; 2016).

These new algorithms have several important benefits and contributions. The first contribution is that, according to this study, we run the selective sampling algorithm over missing values without any preprocessing procedure and imputations. The second and more general contribution is that using ensemble clustering brings us two benefits; the first one is that it allows us to represent the incomplete dataset as complete categorical data. The second one is that the performance of LSS using ensemble clustering is usually better than LSS using the Euclidean distance. From the experiments we conclude that our method is more appropriate for selecting the most informative samples for datasets with missing values.

The proposed methods also have disadvantages. The first algorithm that integrates the MD_E distance function within the framework of the original LSS algorithm has two limitations:

(i) The MDE distance is equivalent to the Euclidean distance and we show in previous work over complete datasets that this distance function might not reflect the actual similarity between the objects.

(ii) After selecting the samples, the data still contain missing values and we should use classifiers that can deal with missing values. In our case we use the kNN that we developed to deal with missing values, but we cannot use the SVM or decision trees classifiers, for example.

As a result, we decided to develop algorithm that uses the ensemble clustering and solves these two limitations. This algorithm also has some limitations:

(i) It is less efficient, because it needs to run clustering algorithms first and then LSS.

(ii) It assumes that each equivalence class is pure, which is not satisfied in each class, especially when data are incomplete.

(iii) It is based on the results of clustering algorithms that were developed to deal with missing values.

Another problem with these methods is that at each iteration only a single point is selected to be labeled by the expert. This is especially problematic when the training process is time consuming; the algorithm will not be efficient. As a result, the classification model has to be retrained after each labeled example is requested.

Moreover, for each selected point from the data the application has to retrieve this example and bring it to the expert in a form which can be used for classification. As a result, in our future work we intend to develop a new batch mode active learning method that selects a group of k points at once and gives them to the expert to be classified. This method cannot use the result of the expert's classification of the previous points to help in selecting the next point in the same batch, but has to take it into account.

Acknowledgment

This research was supported by the 450mm Consortiums of the Ministry of Industry and Commerce as well as Applied Materials Company.

References

Abdallah, L. and Shimshoni, I. (2013). An ensemble-clustering-based distance metric and its applications, *International Journal of Business Intelligence and Data Mining* **8**(3): 264–287.

Abdallah, L. and Shimshoni, I. (2014). Mean shift clustering algorithm for data with missing values, *14th International Conference of DaWaK, Munich, Germany*, pp. 426–438.

Abdallah, L. and Shimshoni, I. (2016). k-means over incomplete datasets using mean Euclidean distance, *12th International Conference on Machine Learning and Data Mining, New York, NY*, pp. 113–127.

Bai, X., Zhang, M., Wu, Q., Zheng, R., Zhao, H. and Wei, W. (2015). A novel data filling algorithm for incomplete information system based on valued limited tolerance relation, *International Journal of Database Theory and Application* **8**(6): 149–164.

Clark, P.G., Grzymala-Busse, J.W. and Rzasa, W. (2013). Consistency of incomplete data, *2nd International Conference on Data Technologies and Applications, Marrakech, Morocco*, pp. 80–87.

Clustering datasets (2008). `http://cs.joensuu.fi/sipu/datasets/`, University of Eastern Finland, Joensuu.

Dasgupta, S. and Hsu, D. (2008). Hierarchical sampling for active learning, *25th International Conference on Machine Learning, Helsinki, Finland*, pp. 208–215.

Dekel, O., Gentile, C. and Sridharan, K. (2012). Selective sampling and active learning from single and multiple teachers, *Journal of Machine Learning Research* **13**(1): 2655–2697.

Donders, A.R.T., van der Heijden, G.J., Stijnen, T. and Moons, K.G. (2006). Review: A gentle introduction to imputation of missing values, *Journal of Clinical Epidemiology* **59**(10): 1087–1091.

Grzymala-Busse, J. and Hu, M. (2001). A comparison of several approaches to missing attribute values in data mining, *in* W. Ziarko *et al.* (Eds.), *Rough Sets and Current Trends in Computing*, Springer, Berlin/Heidelberg, pp. 378–385.

Grzymala-Busse, J.W. (2006). A rough set approach to data with missing attribute values, *in* J.F. Peters and Y. Yao (Eds.), *Rough Sets and Knowledge Technology*, Springer, Berlin/Heidelberg, pp. 58–67.

Hospedales, T.M., Gong, S. and Xiang, T. (2013). Finding rare classes: Active learning with generative and discriminative models, *IEEE Transactions on Knowledge and Data Engineering* **25**(2): 374–386.

Ibrahim, J.G., Chen, M.-H., Lipsitz, S.R. and Herring, A.H. (2005). Missing-data methods for generalized linear models: A comparative review, *Journal of the American Statistical Association* **100**(469): 332–346.

Lewis, D. and Gale, W. (1994). A sequential algorithm for training text classifiers, *17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland*, pp. 3–12.

Li, H., Shi, Y., Liu, Y., Hauptmann, A.G. and Xiong, Z. (2012). Cross-domain video concept detection: A joint discriminative and generative active learning approach, *Expert Systems with Applications* **39**(15): 12220–12228.

Lindenbaum, M., Markovitch, S. and Rusakov, D. (2004). Selective sampling for nearest neighbor classifiers, *Machine Learning* **54**(2): 125–152.

Little, R.J. (1988). Missing-data adjustments in large surveys, *Journal of Business & Economic Statistics* **6**(3): 287–296.

Little, R.J. and Rubin, D.B. (2014). *Statistical Analysis with Missing Data*, John Wiley & Sons. Hoboken, NJ.

Lughofer, E. (2012). Hybrid active learning for reducing the annotation effort of operators in classification systems, *Pattern Recognition* **45**(2): 884–896.

MacQueen, J.B. (1967). Some methods for classification and analysis of multivariate observations, *5th Symposium on Math, Statistics, and Probability, Berkeley, CA, USA,* pp. 281–297.

Magnani, M. (2004). Techniques for dealing with missing data in knowledge discovery tasks, *Obtido* **15**(01): 2007.

Nowicki, R.K. (2010). On classification with missing data using rough-neuro-fuzzy systems, *International Journal of Applied Mathematics and Computer Science* **20**(1): 55–67, DOI: 10.2478/v10006-010-0004-8.

Nowicki, R.K., Nowak, B.A. and Woźniak, M. (2016). Application of rough sets in k nearest neighbours algorithm for classification of incomplete samples, *in* S. Kunifuji *et al.* (Eds.), *Knowledge, Information and Creativity Support Systems*, Springer, Berlin/Heidelberg, pp. 243–257.

Stefanowski, J. and Tsoukias, A. (2001). Incomplete information tables and rough classification, *Computational Intelligence* **17**(3): 545–566.

Strehl, A. and Ghosh, J. (2002). Cluster ensembles—A knowledge reuse framework for combining multiple partitions, *Journal of Machine Learning Research* **3**: 583–617.

Tan, M. and Schlimmer, J. (1990). Two case studies in cost-sensitive concept acquisition, *8th National Conference on Artificial Intelligence, Boston, MA, USA*, pp. 854–860.

Turney, P. (1995). Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm, *Journal of Artificial Intelligence Research* **2**(1): 369–409.

Xu, Z., Akella, R. and Zhang, Y. (2007). Incorporating diversity and density in active learning for relevance feedback, *in* G. Amati *et al.* (Eds.), *Advances in Information Retrieval*, Springer, Berlin/Heidelberg, pp. 246–257.

Zhang, S., Qin, Z., Ling, C. and Sheng, S. (2005). Missing is useful: Missing values in cost-sensitive decision trees, *IEEE Transactions on Knowledge and Data Engineering* **17**(12): 1689–1693.

Zhang, Y., Wen, J., Wang, X. and Jiang, Z. (2014). Semi-supervised learning combining co-training with active learning, *Expert Systems with Applications* **41**(5): 2372–2378.

Loai Abdallah received his B.Sc. in mathematics and management information systems and his M.Sc. and Ph.D. in mathematics, all from the University of Haifa. Loai has been a member of the Department of Mathematics and Computer Science at the College of Sakhnin and the Department of Community Information Systems at Zefat Academic College since 2011. His current research interest is in data mining.

Ilan Shimshoni received his B.Sc. in mathematics from the Hebrew University in Jerusalem, his M.Sc. in computer science from the Weizmann Institute of Science, and his Ph.D. in computer science from the University of Illinois at Urbana Champaign (UIUC). Ilan was a post-doctorate fellow at the Faculty of Computer Science at Technion (1995–1998), and a member of the Faculty of Industrial Engineering and Management (1998–2005). He joined the Department of Information Systems (IS) at Haifa University in October 2005.

Received: 15 October 2015
Revised: 3 March 2016
Re-revised: 2 May 2016
Accepted: 19 July 2016