Exploring Early Classification Strategies of Streaming Data with Delayed Attributes

Mónica Millán-Giraldo[1], J. Salvador Sánchez[1], and V. Javier Traver[1]

Dep. Lenguajes y Sistemas Informáticos and
Institute of New Imaging Technologies (http://init.uji.es),
Universitat Jaume I, 12071, Castellón (Spain)

Abstract. In contrast to traditional machine learning algorithms, where all data are available in batch mode, the new paradigm of streaming data poses additional difficulties, since data samples arrive in a sequence and many hard decisions have to be made on-line. The problem addressed here consists of classifying streaming data which not only are unlabeled, but also have a number l of attributes arriving after some time delay τ. In this context, the main issues are what to do when the unlabeled incomplete samples and, later on, their missing attributes arrive; when and how to classify these incoming samples; and when and how to update the training set. Three different strategies (for $l = 1$ and constant τ) are explored and evaluated in terms of the accumulated classification error. The results reveal that the proposed on-line strategies, despite their simplicity, may outperform classifiers using only the original, labeled-and-complete samples as a fixed training set. In other words, learning is possible by properly tapping into the unlabeled, incomplete samples, and their delayed attributes. The many research issues identified include a better understanding of the link between the inherent properties of the data set and the design of the most suitable on-line classification strategy.

Key-Words: Data mining; Streaming data; On-line classification; Missing attributes

1 Introduction

Most of traditional learning algorithms assume the availability of a training set of labeled objects (examples or instances) in memory. In recent years, however, advances in information technology have lead to a variety of applications in which huge volumes of data are collected continuously, thus making impossible to store all data, or process any particular object more than once. Under these circumstances, data are not available as a batch but comes one object at a time (called *streaming data*). In general, a data stream is defined as a sequence of instances [2, 9]. Data streams differ from the conventional model in important elements [3] that bring new challenges: (i) The objects in the stream arrive on-line; (ii) The system has no control over the order in which incoming data arrive to be processed; (iii) Data streams are potentially unbounded in size.

Classification is perhaps the most widely studied problem in the context of data stream mining. Although substantial progress has been made on this topic [1, 8, 13], a number of issues still remain open. For example, many classification models do not

make adequate use of the history of data streams in order to accommodate changes in class distribution (known as *concept drift*). The scenario we consider in this paper faces a new problem that may appear in several real-world applications. We assume that each object of a data stream is a vector of d attribute values without a class label. The aim of the classification model is to predict the true class of each incoming object as soon as possible (ideally, in real time). However, suppose that the attribute values are obtained from different sensors. These may produce that the attribute values will become available at different times if some sensor requires more processing time to compute an attribute value than the others or even, if some sensors fail. Therefore we are considering the problem of classifying streaming data where one or more attributes arrive with a delay. As an example, when a sensor fails in a production process, it might not be feasible to stop everything and in this case, the system should employ the information available at present time. Three main issues here are: (i) How to classify the incoming sample with missing attributes; (ii) Whether to update the training (reference) set after predicting the class label of an incomplete object or wait until the attribute vector has been completed; and (iii) What to do when the missing attributes arrive.

In the literature, there exist many algorithms for handling data with missing attributes in off-line learning [4, 6, 7, 10, 16], but no one is absolutely better than the others. The most representative categories of these are:

1. Removing examples with missing attributes: The simplest way of dealing with missing values is to discard the examples that contain the missing values. Due to its simplicity, this technique may lose relevant information.
2. Projection: The l missing attributes are ignored. This implies to map the d dimensional input vectors onto an $(d - l)$ instance space.
3. Imputation: It tries to guess the missing values. In fact, usually missing values depend on other values, and if we find a correlation between two attributes, we may use it to impute missing items. Imputations may be deterministic or random (stochastic). In the first case, imputations are determined by using the complete data, and are the same if the method is applied again. In the second case, imputations are randomly drawn.

Despite the problem of missing attributes has been widely studied in off-line learning, to the best of our knowledge it has not previously been considered in the context of on-line learning with streaming data, which makes the problem considerably more challenging. This paper reports a preliminary study of three straightforward strategies for an early classification of streaming data with missing attributes. By *early* we mean that classification of an incoming object is done *before* the whole attribute vector is known. Many applications can benefit from performing this early classification, since there may be some kind of loss associated with waiting for the missing attributes to arrive. In the present work we concentrate on the case of a single missing attribute which happens to be the same and arrive with a constant delay.

2 Classification of Streaming Data with Delayed Attributes

At time step t in the scenario of attributes arriving with a delay, we have a reference set S_t (a set of labeled examples with all attributes available). Then, a new unlabeled

object $\mathbf{x}_{t+1}$ with one missing attribute $x_{t+1}^{(i)}$ arrives. After predicting the label for $\mathbf{x}_{t+1}$, the system receives the value of the attribute $x_{t-\tau+1}^{(i)}$ corresponding to the object that came τ steps earlier, $\mathbf{x}_{t-\tau+1}$. Therefore, objects from $\mathbf{x}_{t-\tau+2}$ to $\mathbf{x}_{t+1}$ are still with one missing attribute.

Here, one key question is whether to use the unlabeled data with missing attributes to update the reference set S_t and in such a case, how to do it. In addition, we have to decide how to best utilize the value of the missing attribute $x_t^{(i)}$ when this becomes available.

When a new unlabeled object $\mathbf{x}_{t+1}$ arrives, the system has to provide a prediction for its label based on the information available up to time t. In this situation, it would be desirable to make use of the *confidence* with which the previous classifications have been made. That is why a modification of the k-Nearest Neighbor (k-NN) rule [15] is here used, since its stochastic nature results suitable to properly manage the confidence measurements. On the other hand, for handling the missing attribute of object $\mathbf{x}_{t+1}$, we employ the projection strategy because of its simplicity and its proven good behavior.

2.1 A Classifier with Confidence Measurements

All instances in the reference set have a confidence value for each class, indicating the probability of belonging to the corresponding class. When a new unlabeled object $\mathbf{x}_{t+1}$ from the data stream arrives, its confidence values (one per class) are estimated. Thus the object will be assigned to the class with the highest confidence value.

To estimate the confidence values of the incoming object $\mathbf{x}_{t+1}$, its k nearest neighbors from the reference set S_t are used. The confidences of its k nearest neighbors, which contribute a weight by each class to the object $\mathbf{x}_{t+1}$, and the distances between them and the new object $\mathbf{x}_{t+1}$ are also employed.

More formally, let k be the number of nearest neighbors, let $\mathbf{n}_j$ be the *j*-th nearest neighbor of $\mathbf{x}_{t+1}$, let $p_m(\mathbf{n}_j)$ denote the confidence (probability) that the *j*-th nearest neighbor belongs to class m, and let $d(\mathbf{x}_{t+1}, \mathbf{n}_j)$ be the Euclidean distance between the object $\mathbf{x}_{t+1}$ and $\mathbf{n}_j$. The confidence of the object $\mathbf{x}_{t+1}$ in relation with the class m, say $P_m(\mathbf{x}_{t+1})$, is given by the following equation [15]:

$$P_m(\mathbf{x}_{t+1}) = \sum_{j=1}^{k} p_m(\mathbf{n}_j) \frac{1}{\epsilon + d(\mathbf{x}_{t+1}, \mathbf{n}_j)}, \tag{1}$$

where ϵ is a constant value ($\epsilon = 1$), which is employed to avoid uncertain values in the division when the object $\mathbf{x}_{t+1}$ is very similar or very close to its *j*-th nearest neighbor.

The above expression states that the confidence that an object $\mathbf{x}_{t+1}$ belongs to a class m is the weighted average of the confidences that its k nearest neighbors belong to class m. The weight is inversely proportional to the distance from the object to the corresponding k nearest neighbors. In order to get a proper probability, the confidence $P_m(\mathbf{x}_{t+1})$ in Eq. (1) is divided by the sum of the confidences of the k nearest neighbors to all the classes:

$$p_m(\mathbf{x}_{t+1}) = \frac{P_m(\mathbf{x}_{t+1})}{\sum_{r=1}^{c} P_r(\mathbf{x}_{t+1})}, \tag{2}$$

where $p_m(\mathbf{x}_{t+1})$ is the normalized confidence (called *the posterior probability*) of the object $\mathbf{x}_{t+1}$, c is the number of classes, and $P_r(\mathbf{x}_{t+1})$ is the confidence of the object $\mathbf{x}_{t+1}$ to belong to class r.

As the objects of the reference set S_t are labeled elements, their confidence values were initially set to 1 for the true class (the class to which they belonged), and zero for the remaining classes. During the on-line learning, the confidence of all new objects incorporated into the training set will be updated according to the probability values of Eq. (2).

2.2 Managing Incomplete Objects and Their Delayed Attribute

Assuming that at step t we have a reference set S_t available, on-line classification of incomplete streaming data consists of three main elements: (i) The technique to handle the situation of a missing attribute $x_{t+1}^{(i)}$ of the new unlabeled object $\mathbf{x}_{t+1}$; (ii) The classifier to predict the class label for this object; and (iii) The strategy to manage the new information derived from the value of the attribute $x_{t+1}^{(i)}$ when it arrives τ steps later.

Regarding the first issue, as stated before, the projection strategy is used: the arriving object as well as those in the reference set are simply mapped onto the $d-1$ dimensional space. Second, as for the prediction of the class label for $\mathbf{x}_{t+1}$, the k-NN classifier based on posterior probabilities (Sect. 2.1), is used. Finally, since it is not obvious which is the best way to profit from the new information gained with the arrival of the attribute $x_{t-\tau+1}^{(i)}$ at time step $t+1$, three different strategies are explored:

1. *Do-nothing*: This is a *passive* strategy where, while the incoming object is incorporated into the current reference set S_t, nothing is done when the value of the missing attribute $x_{t-\tau+1}^{(i)}$ arrives after τ time steps. However, the attribute value of the corresponding object, $\mathbf{x}_{t-\tau+1}$, is set to the value $x_{t-\tau+1}^{(i)}$.
2. *Put and reclassify*: This is a *proactive* strategy differing from the *do-nothing* strategy in that the object $\mathbf{x}_{t-\tau+1}$ is also reclassified, this time using *all* attributes.
3. *Wait and classify*: This is a *reactive* strategy where, unlike the two previous strategies, the new object $\mathbf{x}_{t+1}$ is *not* included in the reference set S_t *until* its missing attribute is received after τ time steps. Only by then, the complete object is classified and incorporated into the reference set $S_{t+1+\tau}$.

The different nature of these strategies will allow to gain some insight into which may be the best way to proceed in the context of on-line classification of streaming data with missing (but delayed) attributes. This will also provide cues on what further research avenues to follow. The assessment of the different strategies proposed has been done on extensive experimental work, which is subsequently presented.

3 Experiments and Results

The experiments here carried out are directed to empirically evaluate each strategy described in the previous section, pursuing to determine which of these is the most suitable

for the classification of incomplete streaming data. The ultimate purpose of this preliminary study is to investigate whether the employment of attribute values that arrive with a delay allows to improve the system performance or not.

Table 1. Characteristics of the real data sets used in the experiments.

Data set	Features	Classes	Objects	Reference Set	Data Stream	Source
iris	4	3	150	12	138	UCI[1]
wine	13	3	178	39	139	UCI
crabs	6	2	200	12	188	Ripley[2]
sonar	60	2	208	120	88	UCI
laryngeal1	16	2	213	32	181	Library[3]
thyroid	5	3	215	15	200	UCI
breast	9	2	277	18	259	UCI
intubation	17	2	302	34	268	Library
heart	13	2	303	26	277	UCI
ecoli	7	8	336	56	280	UCI
liver	6	2	345	12	333	UCI
spect	44	2	349	88	261	Library
voice9	10	9	428	90	338	Library
wbc	30	2	569	60	509	UCI
palynomorphs	31	3	609	93	516	Private[4]
australian	42	2	690	84	606	UCI
laryngeal2	16	2	692	32	660	Library
pima	8	2	768	16	752	UCI
vehicle	18	4	846	72	774	UCI
vowel	11	10	990	110	880	UCI
german	24	2	1000	48	952	UCI
image	19	7	2310	133	2177	UCI

[1]UCI [14]
[2]Ripley [11]
[3]Library `http://www.bangor.ac.uk/~mas00a/activities/real_data.htm`
[4]Images of pieces of kerogen extracted from microscope images of palynomorphs

Experiments were conducted as follows:

Data sets: Twenty-two real data sets (summary of whom is given in Table 1) were employed in the experiment. Data were normalized in the range $[0, 1]$ and all features were numerical. In the table, the data sets are sorted by increasing size.

Partitions: For each database, 10 runs were carried out. A random stratified sample of $d \times c$, being d the number of attributes and c the number of classes, was taken as the initial labeled references set S_0. The remaining part of each database was used as the incoming on-line streaming data. To simulate independent and identically-distributed sequences, the data were shuffled before each of these 10 runs.

Incomplete objects: A new object with one missing attribute from the on-line data was fed to the system at a time step. Both the most and the least relevant attributes

of each database were simulated to be missing. Attribute relevance was estimated by means of the Jeffries-Matusita distance [5].

Delay: The missing attribute comes after $\tau = 5$ time steps. When the delayed attribute arrives, the corresponding object is completed with the true attribute value.

Classification: At each time step t, the respective strategy to handle delayed attributes was applied. The accumulated classification error (the total number of misclassifications divided by the number of samples processed up to t) was computed. In this way we created a progression curve (trend line), which is the classification error as a function of the number of on-line objects seen by the classifier. The results were averaged across the 10 runs giving a single progression curve for each data set.

For each of the 10 runs of the experiment, all strategies received the same partitions of the data into initial labeled reference set and streaming data set. These on-line data were presented to all methods in the same order so that performance differences can not be attributable to different data (order).

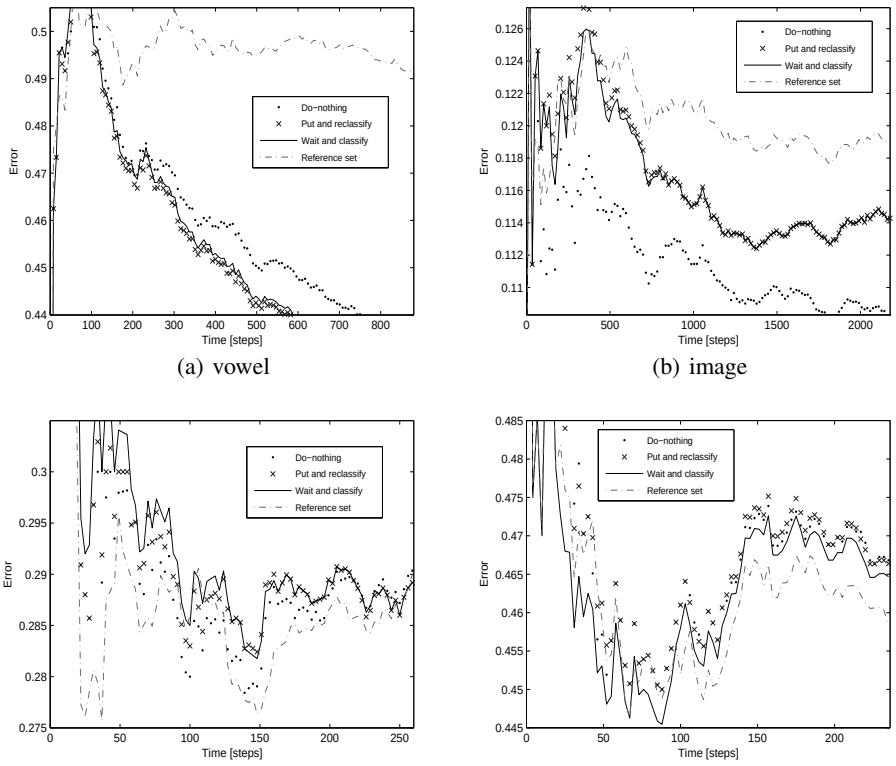
3.1 Results

Table 2 reports the average errors estimated across all incoming objects for each strategy. To evaluate whether the performance improves at all with streaming data, we have also included the error using only the initial reference set S_0. For each database, the first row corresponds to the classification errors when the least relevant attribute arrives with a delay. The second row is for the most relevant attribute. Highlighted in bold are the results being better than those obtained by using only the initial reference set for classification. Underlined values correspond to the best strategy for each database and each attribute.

As can be seen, out of the 22 databases, the strategies here proposed give better results than using the initial reference set on 14 cases when the missing attribute corresponds to the most relevant, and on 12 for the least relevant attribute. Performance differences are small among the proposed strategies as well as between each of them and the baseline case.

Detailed results for four databases are provided in the plots in Fig. 1, with the x and y axes representing, respectively, the number of objects fed to the system at each time step, and the accumulated classification error averaged over the 10 runs. The results on the *vowel* database (Fig. 1(a)) are very interesting, since all the proposed strategies outperform the baseline case (which uses only the full and labelled samples in the initial reference set). Furthermore, the accumulated classification error decreases over time, which is a clear evidence of how the system is learning from the incoming, incomplete and unlabeled samples. Finally, in this case, where the delayed attribute was the one with the most relevance, the strategies *Put and reclassify* and *Wait and classify* can be seen to work better than *Do-nothing*. A likely explanation for this behavior is that, since the attribute is important for the correct classification, it is worth waiting for the delayed attribute to arrive either for reclassifying the object (*Put and reclassify*), or for incorporating the object into the reference set only once it is complete (*Wait and classify*).

Results for the *image* database (Fig. 1(b)) illustrate again how the considered strategies can boost the classification performance with respect to the conservative baseline approach. Interestingly, it is the *Do-nothing* strategy which now behaves better than the other two. Since the delayed attribute was the least relevant, it might happen that this attribute is hindering the classifier rather than helping it. As a consequence, and in the context of the *projection* technique that is being used in this work, it turns out to be better to passively ignore the attribute when it arrives than trying to make the most of it.



(a) vowel (b) image

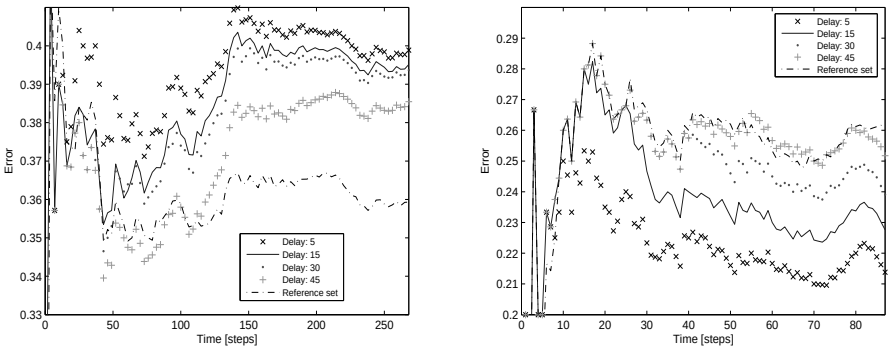
(c) spect (d) heart

Fig. 1. Average error computed for ten runs using the *Do-nothing*, *Put and reclassify* and *Wait and classify* strategies. The baseline case (i.e., using only the initial reference set) is included for comparison with an off-line strategy. The delayed attribute was either the most (a, c) or the least relevant (b, d)

Figs. 1(c,d) provide two examples where the performance of the compared strategies was similar to or slightly worse than the baseline case when the delayed attribute is the most (Figs. 1(c)) or the least (Fig. 1(d)) relevant. To explain the rationale behind these differences in performance, a good insight must be gained into what makes these databases different. This understanding will help us devise more general and robust strategies.

While in all examples above the missing attribute was delayed $\tau = 5$ time steps, it is interesting to evaluate how the actual delay affects the performance of the strategies under analysis. To this end, the same testing procedure was repeated for $\tau \in \{5, 15, 30, 45\}$ for several data sets. It was found that the *Do-nothing* and *Put and reclassify* strategies did not exhibit a significant performance difference for distinct delays. However, differences were observed for the *Wait and classify* strategy, as illustrated in Fig. 2 for two of the tested databases.

In those data sets where this strategy did not work well, such as *intubation* (Fig. 2(a)), the accumulated classification error decreased when the delay increased. This can be explained as follows: the missing attribute happens to be unimportant (even harmful) for the classification and therefore, the longer it takes the attribute to arrive, the longer it takes to be incorporated into the object (and then into the training set) and thereby, less time it is affecting in the classification of subsequent objects. However, in cases such as *sonar* (Fig. 2(b)), where this strategy tends to work well, the more the delay, the higher the error. In this situation, the missing attribute appears to be necessary for the correct prediction of incoming objects.



(a) intubation (b) sonar

Fig. 2. Average accumulated classification errors computed for ten runs over two databases, using the *Wait and classify* strategy, when the delayed attribute arrived $\tau = 5$, $\tau = 15$, $\tau = 30$ and $\tau = 45$ time steps after the incomplete object. In both figures the delayed attribute corresponds to the most relevant

4 Conclusions and Further Extensions

In this paper, we have explored three simple strategies for the classification of streaming data with a single missing attribute. More specifically, we have presented a preliminary study for handling on-line data where the complete attribute vector arrives with a constant delay. Despite their simplicity, the results of the three strategies have shown some gains in performance when compared to the use of the initial reference set. Although these benefits are still marginal, the most important finding is that it seems possible to

Table 2. Average errors estimated across all incoming objects

Data set	Initial reference set	Do-nothing	Put and reclassify	Wait and classify
iris	0.1076	**0.0954**	**0.0919**	**0.0893**
	0.1236	**0.1134**	**0.1111**	**0.1093**
wine	0.0473	0.0515	0.0515	0.0476
	0.0483	0.0550	0.0577	0.0555
crabs	0.4695	**0.4332**	**0.4226**	**0.4313**
	0.4875	**0.4341**	**0.4334**	**0.4456**
sonar	0.2299	**0.2205**	**0.2206**	**0.2207**
	0.2319	**0.2225**	**0.2223**	**0.2239**
laryngeal1	0.1974	0.2033	0.2044	0.2051
	0.1931	**0.1915**	**0.1915**	0.1957
thyroid	0.1860	**0.1639**	**0.1751**	**0.1763**
	0.2531	**0.2336**	**0.2158**	**0.2168**
breast	0.3108	0.3287	0.3226	0.3196
	0.3010	0.3284	0.3279	0.3285
intubation	0.3214	0.3714	0.3631	0.3655
	0.3581	0.3899	0.3859	0.3838
heart	0.4641	0.4704	0.4707	0.4653
	0.4731	0.4809	0.4780	0.4732
ecoli	0.2060	0.2146	0.2115	0.2120
	0.1834	0.1841	**0.1786**	**0.1782**
liver	0.4689	**0.4610**	**0.4607**	**0.4648**
	0.4732	**0.4686**	**0.4631**	**0.4695**
spect	0.2862	**0.2852**	0.2883	0.2925
	0.2895	0.2928	0.2943	0.2955
voice9	0.6366	**0.6352**	**0.6333**	0.6368
	0.6428	**0.6407**	0.6430	0.6447
wbc	0.0427	**0.0417**	**0.0414**	**0.0423**
	0.0467	**0.0432**	**0.0411**	**0.0428**
palynomorphs	0.1878	0.1904	0.1902	0.1890
	0.1974	0.2003	0.1985	**0.1966**
australian	0.1590	**0.1570**	**0.1576**	**0.1589**
	0.1661	0.1710	0.1714	0.1730
laryngeal2	0.0545	**0.0538**	**0.0533**	**0.0526**
	0.0538	**0.0526**	**0.0524**	**0.0516**
pima	0.3120	0.3290	0.3271	0.3287
	0.3415	**0.3383**	**0.3353**	**0.3400**
vehicle	0.4451	**0.4336**	**0.4334**	**0.4368**
	0.4383	**0.4366**	**0.4337**	**0.4339**
vowel	0.4410	**0.4180**	**0.4162**	**0.4169**
	0.4960	**0.4608**	**0.4530**	**0.4541**
german	0.3156	0.3225	0.3216	0.3202
	0.3367	0.3434	**0.3350**	**0.3334**
image	0.1204	**0.1117**	**0.1169**	**0.1166**
	0.1410	**0.1269**	**0.1254**	**0.1251**

design some method to consistently handle the incomplete data in on-line classification of data streams.

The ultimate purpose of this work was to describe a novel and relevant problem that can be present in many real-world applications. Our study has revealed a number of interesting research directions regarding the classification of streaming (and incomplete) unlabeled data, such as: (i) An analysis of how the relevance of the missing attribute affects the different strategies; (ii) The design of more elaborated methods for early classification of streaming data; (iii) The study of the benefits of different techniques for handling missing attributes; (iv) An analysis of the case where the environment does change with time, and the reference sets will have to track these changes; and (v) An exploration of a more general situation with more than one delayed attribute and varying time delays.

References

1. Agarwal, C.: On-Demand Classification of Data Streams. In: Proc. ACM International Conference on Knowledge Discovery and Data Mining, pp. 503–508 (2004).
2. Agarwal, C.: Data Streams: Models and Algorithms. Springer, New York (2007).
3. Babcock, B., Babu, S., Datar, M, Motwani, R., Widom, J.: Models and Issues in Data Stream Systems. In: Proc. 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 1–16 (2002).
4. Batista, G.E.A.P.A., Monard, M.D.: An Analysis of Four Missing Data Treatment Methods for Supervised Learning. Applied Artificial Intelligence, vol. 17(5), pp. 519–533 (2003).
5. Bruzzone, L., Roli, R., Serpico, S.B.: An extension of the Jeffreys–Matusita distance to multiclass cases for feature selection. IEEE Trans. on Geoscience and Remote Sensing, vol. 33(6), pp. 1318–1321 (1995).
6. Delavallade, T., Dang, T.H.: Using Entropy to Impute Missing Data in a Classification Task. In: Proc. IEEE International Conference on Fuzzy Systems, pp. 577–582 (2007).
7. Friedman, J.H., Kohavi, R., Yun, Y.: Lazy Decision Trees. In: Proc. 13th National Conference on Artificial Intelligence, pp. 717–724 (1996).
8. Ganti, V., Gehrke, J., Ramakrishnan, R.: Demon: Mining and Monitoring Evolving Data. IEEE Trans. on Knowledge and Data Engineering, vol. 13(1), pp. 50–63 (2001).
9. Muthukrishnan, S.: Data Streams: Algorithms and Applications. Foundations and Trends in Theoretical Computer Science, pp. 117–236 (2005).
10. Quinlan, J.: Induction of Decision Trees. Machine Learning, vol. 1(1), pp. 81–106 (1986).
11. Ripley, B.D.: Pattern Recognition and Neural Networks. Cambridge University Press (1996).
12. Little, R.J.A., Rubin, D.B.: Statistical Analysis with Missing Data. Wiley, New York (1987).
13. Street, W.N., Kim, Y: A Streaming Ensemble Algorithm (SEA) for Large-Scale Classification. In: Proc. 7th International Conference on Knowledge Discovery and Data Mining, pp. 377–382 (2001).
14. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository, School of Information and Computer Science, University of California, Irvine, CA (2007) `http://archive.ics.uci.edu/ml/`.
15. Vázquez, F., Sánchez, J.S., Pla, F.: A Stochastic Approach to Wilsons Editing Algorithm. In: Proc. 2nd Iberian Conference on Pattern Recognition and Image Analysis, pp. 35–42 (2005).
16. Zhang, S.C., Qin, Y.S., Zhu, X.F., Zhang, J.L., Zhang, C.Q.: Optimized Parameters for Missing Data Imputation. In: Proc. 9th Pacific–Rim International Conference on Artificial Intelligence, pp. 1010–1016 (2006).