International Journal of Approximate Reasoning 69 (2016) 35–57

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Online streaming feature selection using rough sets

S. Eskandari *, M.M. Javidi

Shahid Bahonar University of Kerman, Kerman, Iran

ARTICLE INFO

Article history:
Received 21 April 2015
Received in revised form 9 October 2015
Accepted 10 November 2015
Available online 14 November 2015

Keywords:
Feature selection
Online streaming feature selection
Rough sets theory
Significance

ABSTRACT

Feature Selection (FS) is an important pre-processing step in data mining and classification tasks. The aim of FS is to select a small subset of most important and discriminative features. All the traditional feature selection methods assume that the entire input feature set is available from the beginning. However, online streaming features (OSF) are an integral part of many real-world applications. In OSF, the number of training examples is fixed while the number of features grows with time as new features stream in. A critical challenge for online streaming feature selection (OSFS) is the unavailability of the entire feature set before learning starts. Several efforts have been made to address the OSFS problem, however they all need some prior knowledge about the entire feature space to select informative features. In this paper, the OSFS problem is considered from the rough sets (RS) perspective and a new OSFS algorithm, called OS-NRRSAR-SA, is proposed. The main motivation for this consideration is that RS-based data mining does not require any domain knowledge other than the given dataset. The proposed algorithm uses the classical significance analysis concepts in RS theory to control the unknown feature space in OSFS problems. This algorithm is evaluated extensively on several high-dimensional datasets in terms of compactness, classification accuracy, run-time, and robustness against noises. Experimental results demonstrate that the algorithm achieves better results than existing OSFS algorithms, in every way.

1. Introduction

Although a larger feature set gives more information about a concept, in the supervised classification and learning tasks we need to keep the feature set as small as possible to reduce computational complexity, gain good generalization performance and increase accuracy [16,47]. In the classical feature selection problem, a dataset of m feature vectors consisting of n input features and one or more output feature is given. The task of feature selection is to select the smallest subset of the most important and discriminative input features. A variety of feature selection algorithms have been developed during the last three decades [27,34,1,38,11,6,39,55,3,33,46,54,5,21].

All the traditional feature selection methods assume that the entire input feature set is available from the beginning. However, online streaming features (OSF) is an integral part of many real-world applications. In OSF, the number of feature vectors is fixed while feature set grows with time. There are two major reasons for OSF:

1. The feature space is unknown or even infinite. For example, in bioinformatic and clinical machine learning problems, acquiring the entire set of features for every training instance is expensive due to the high cost of lab experiments

* Corresponding author.
E-mail addresses: eskandari@math.uk.ac.ir (S. Eskandari), javidi@uk.ac.ir (M.M. Javidi).

http://dx.doi.org/10.1016/j.ijar.2015.11.006
0888-613X/

[51]. As another example, in texture-based image segmentation problems, the number of different texture filters can be infinite and therefore acquiring the entire feature set is infeasible [40,53,15]. In all these scenarios, we need to incrementally update the feature set as new features are available over time.

2. The feature space is known but feature streaming offers many advantages. This scenario is common when the feature space is very large, which makes exhaustive search over the entire feature space expensive or even infeasible. For example, a dataset in document classification problems may contain a set of hundreds of thousands of potential features [48] and therefore, considering a batch feature selection algorithm over the entire dataset needs considerable storage and computational time capabilities.

Online streaming feature selection (OSFS) is the task of selecting a best feature subset from so-far-seen features in OSF. Any OSFS method must satisfy three critical conditions; First, it should not require any domain knowledge about feature space, because the full feature space is unknown or inaccessible. Second, it should allow efficient incremental updates in selected features. We usually have a fixed, limited amount of computational time available between each feature arrival, and so we want to use a method whose update time does not increase without limit as more features are seen [40]. Third, it should be as accurate as possible at each time instance to allow having reliable classification and learning tasks at that time instance.

We would like to distinguish OSFS in this work from the previous studies of incremental feature selection (IFS) in [51,19,32,20,31,28]. In IFS, which is also known as standard online feature selection (SOFS) [51,19] and dynamic feature selection (DFS) [31], the number of training instances (feature vectors) grows with time while the number of attributes (features) is assumed to be fixed. IFS is common in monitoring environments using sensor networks such as network traffic monitoring for net controlling applications, streams of stock data reported from various stock exchanges monitoring for financial analysis applications, and query monitoring in an environment like the Internet. This is in some sense dual to our OSFS scenario in this work where features arrive sequentially and training instances are all available from the beginning.

Although several research efforts have been made to address OSFS [40,48,57,53,50], none of these algorithms satisfy all the critical conditions for an OSFS algorithm. In the paper, the OSFS problem is considered from the rough sets (RS) perspective. The main motivation for this consideration is that RS-based data mining does not require any domain knowledge other than the given dataset. This property seems to be an ideal tool in OSF scenarios. Several successful RS-based feature selection algorithms are proposed in the literatures [18,17,29,30,35,45,56,22,36,26]. However, all these algorithms consider the batch feature selection problem and are not applicable to OSF scenarios. In this paper, a new OSFS algorithm, called OS-NRRSAR-SA, is proposed. This algorithm adopts the classical RS-based feature significance concept to eliminate irrelevant features. The efficiency and accuracy of the proposed algorithm is demonstrated using several experimental results.

The remainder of the paper is organized as follows: Section 2 discusses related works. Section 3 summarizes the theoretical background of RSFS along with a look at the rough set extensions and modifications. Section 4 discusses the new OSFS algorithm. Section 5 reports experimental results and Section 6 concludes the paper.

2. Related work

Traditional feature selection methods can be classified into wrapper, filter and embedded methods. In the wrapper approaches [27,34], the classifier or induction algorithm is a part of feature subset evaluation process. For each subset of input features, a classifier is trained and the subset with minimum classification error is selected. Although this method has high accuracy, the exponential number of subsets makes the method computationally expensive. Moreover, using a classifier to evaluate feature subsets, biases the selected subset to that classifier. In the filter approaches, the feature subset evaluation is independent of the classifier or induction algorithm. For each candidate subset of features, evaluation measures such as information [1,3,38], consistency [11] and relevance [38,53] are applied and the best feature subset is selected. These methods are less accurate than wrapper methods. However, they are simple, fast and unbiased to any special classifier [53,16]. In the embedded approaches [39,55], the feature selection is considered as an integral part of a model training process. In such approaches the trainer tries to make a trade off between model complexity and model accuracy by selecting or removing features. Embedded methods are typically more efficient than the wrappers [53]. However the results are biased to the classifier.

The traditional methods discussed above, examine a batch of all candidate features at each iteration to select the best feature subset. Therefore they consider that all the candidate features are available before starting the feature selection process. Adopting the traditional feature selection methods in OSF scenarios poses critical challenges, because the feature space is not available from the beginning in this scenarios.

Motivated by these challenges, several research efforts have been made to address OSFS. Perkins and Theiler proposed a grafting based algorithm for this problem [40]. Grafting is an iterative gradient descent algorithm, which treats the feature selection task as part of a regularized risk minimization problem [39]. In the online version of this algorithm, a newly seen feature is added to the selected features if the improvement in the model accuracy is greater than a predefined threshold λ. While this algorithm is able to handle streaming features, it is ineffective in dealing with true OSF scenarios for three reasons; 1) choosing a suitable λ requires information about the global feature space. 2) This algorithm suffers from the so-called nesting effect [41]. If a previously chosen feature is later found to be redundant, there is no way for it to be

Table 1
An example table.

$x \in U$	f_1	f_2	f_3	f_4	d
1	1	2	0	2	1
2	0	2	1	1	1
3	1	1	2	2	3
4	2	2	2	3	4
5	2	2	1	3	3
6	1	1	2	2	2
7	0	1	1	1	1
8	2	2	0	3	2

discarded. 3) This algorithm is based on gradient descent optimization which is a numerical optimization approach and therefore it is not applicable to the non-numerical feature spaces.

Ungar et al. [48] proposed a streamwise regression algorithm, called information-investing, for dynamically generated feature streams. In this algorithm, a newly generated feature is added to the model if the entropy reduction is greater than the cost of coding. Zhou et al. [57] proposed α-investing, a very similar algorithm to information-investing, which uses the p-value of the generated feature as a criterion for adding it to the model. In spite of their success in handling feature spaces of unknown or even infinite sizes, these algorithms suffer from the nesting effect, such as grafting. Moreover, these algorithms require prior knowledge about the structure of the feature space to heuristically control the choice of candidate feature selection [53].

Wu et al. [53] proposed a causality-based OSFS algorithm called fast-OSFS. This algorithm contains two major steps, 1) online relevance analysis that discards irrelevant features, and 2) online redundancy analysis, which eliminates redundant features. Although this framework is able to select most informative features in OSF, it uses a conditional independence test which needs a large number of training instances, especially when the number of features contributed in test grows with time. A limited number of training instances is an integral part of many real world datasets and therefore, adopting fast-OSFS in such datasets does not generate reliable results.

Wang et al. [50] proposed a dimension incremental attribute reduction algorithm called DIA-RED. This algorithm maintains a rough sets-based entropy value of the current selected subsets and updates this value whenever new conditional features are added to the dataset. While DIA-RED is able to handle streaming scenarios without any knowledge about feature space, it uses only the information contained within the lower approximation of a set ignoring the information contained in the boundary region. Therefore, this algorithm is not applicable effectively to real-valued datasets. Moreover, DIA-RED does not implement an effective redundant attribute elimination mechanism, and therefore the selected subsets are large during features streaming. This causes ineffective partitioning steps in calculating the rough sets approximations, and therefore, this algorithm is not time efficient for most of the real world datasets.

3. Rough sets

Rough sets theory has been introduced by Pawlak [37] to express vagueness by means of boundary region of a set. The main advantage of this implementation of vagueness is that it requires no human input or domain knowledge other than the given dataset [49,36]. This section describes the fundamentals of the theory. Table 1 will be used as an example dataset to illustrate the corresponding operations.

3.1. Information system and indiscernibility

An information system is a pair $IS = (U, F)$, where U is a non-empty finite set of objects called the universe and F is a non-empty finite set of features such that $f : U \to V_f$, for every $f \in F$. The set V_f is called the value set or domain of f. An information system in rough sets theory is analogous to a dataset in unsupervised machine learning and classification tasks. A decision system is an information system of the form $IS = (U, F, d)$, where d is the decision feature. A dataset in a supervised classification and learning can be considered as a decision system where instances are the objects of universe, features are the elements of F and labels represent decision feature values.

For any set $B \subseteq F \cup \{d\}$, we define the B-indiscernibility relation as:

$$IND_{IS}(B) = \{(x, y) \in U \times U | \forall f \in B, f(x) = f(y)\} \tag{1}$$

If (x, y) belongs to $IND_{IS}(B)$, x and y are said to be indiscernible according to the feature subset B. Equivalence classes of the relation $IND_{IS}(B)$ are denoted $[x]_B$ and referred to as B-elementary sets. The partitioning of U into B-elementary subsets is denoted by $U/IND_{IS}(B)$ or simply U/B. Generating such a partition is a common computational routine, that affects the performance of any rough set based operation represents a general procedure PARTITION to compute U/B. The general procedure PARTITION to compute U/B is displayed in Fig. 1.

The time complexity of PARTITION is $\Theta(|B||P||U|)$, where $|P|$ is the number of generated B-elementary subsets. If none of the objects in U are indiscernible according to B, the number of B-elementary subsets is $|U|$ and therefore the worst-case complexity of PARTITION is $O(|B||U|^2)$.

```
PARTITION(U, B)
U: the universe of objects
B: a subset of features

1: P ← {}
2: for each x in U do
3:     [x]_B ← {x}
4:     for each y in U − {x} do
5:         if ( x(B) = y(B))
6:             [x]_B ← [x]_B ∪ {y}
7:             U ← U − {y}
8:         end if
9:     end for
10:    P ← P ∪ {[x]_B}
11: end for
12: return P
```

Fig. 1. The partitioning algorithm to generate elementary subsets.

For the dataset of Table 1, if $B = \{f_1, f_4\}$, then objects 1, 3, 6 are indiscernible; as are objects 2, 7 and 4, 5, 8. U/B is as follows:

$$U/B = \{\{1, 3, 6\}, \{2, 7\}, \{4, 5, 8\}\}$$

3.2. Lower and upper approximations

Two fundamental concepts of rough sets are the lower and upper approximations of sets. Let $B \subseteq F$ and $X \subseteq U$, the B-lower and B-upper approximations of X are defined as follows:

$$\underline{B}X = \{x | [x]_B \subseteq X\} \tag{2}$$

$$\overline{B}X = \{x | [x]_B \cap X \neq \emptyset\} \tag{3}$$

The $\underline{B}X$ and $\overline{B}X$ approximations define information contained in B [36]. If $x \in \underline{B}X$, it certainly belongs to X but if $x \in \overline{B}X$, it may or may not belong to X. For example, let $B = f_1, f_4$ and $X = \{1, 2, 5, 7, 8\}$, then

$$\underline{B}X = \{2, 7\}$$

$$\overline{B}X = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

By the definition of $\underline{B}X$ and $\overline{B}X$, the objects in U can be partitioned into three parts, called the positive, boundary and negative regions.

$$POS_B(X) = \underline{B}X \tag{4}$$

$$BND_B(X) = \overline{B}X - \underline{B}X \tag{5}$$

$$NEG_B(X) = U - \overline{B}X \tag{6}$$

In the example, the three regions for $B = \{f_1, f_4\}$ and $X = \{1, 2, 5, 7, 8\}$ are as follow:

$$POS_B(X) = \{2, 7\}$$

$$BND_B(X) = \{1, 3, 4, 5, 6, 8\}$$

$$NEG_B(X) = \emptyset$$

3.3. Dependency

Discovering dependencies between attributes is an important issue in data analysis. Let D and C be subsets of $F \cup \{d\}$. For $0 \leq k \leq 1$, it is said that D depends on C in the kth degree (denoted $C \Rightarrow_k D$), if

$$k = \gamma(C, D) = \frac{|POS_C(D)|}{|U|}, \tag{7}$$

QUICKREDUCT(C, d)
C: The set of all conditional features
d: The decision feature

1: $R \leftarrow \{\}$
2: **while** $(\gamma(R, d) \neq \gamma(C, d))$ **do**
3: $\quad x \leftarrow \arg\max_{f \in C-R} (\gamma(R \cup \{f\}, d) - \gamma(R, d))$
4: $\quad R \leftarrow R \cup \{x\}$
5: **end while**
6: **return** R

Fig. 2. The QUICKREDUCT algorithm.

where

$$POS_C(D) = \bigcup_{X \in U/D} \underline{C}X$$

is called a positive region of the partition U/D with respect to C. This region is the set of all elements of U that can be uniquely classified to blocks of the partition U/D, by means of C [45].

In the example, if $C = \{f_1, f_4\}$ then:

$$POS_C(d) = \bigcup \left(\underline{C}\{1, 2, 7\}, \underline{C}\{3, 5\}, \underline{C}\{4\}, \underline{C}\{6, 8\}\right)$$
$$= \{2, 7\}$$

The degree of dependency of attribute d on attributes $\{f_1, f_4\}$ is:

$$\gamma(\{f_1, f_4\}, d) = \frac{|POS_{\{f_1, f_4\}}(d)|}{|U|} = \frac{2}{8}$$

The functional dependency of D and C $(C \Rightarrow D)$ is a special case of dependency where $\gamma(C, D) = 1$. In this case we say that all values of attributes from D are uniquely determined by the values of attributes from C.

3.4. Reduct

A reduct R is a minimal set of features $R \subseteq C$, such that $\gamma(R, d) = \gamma(C, d)$ [22,26]. An optimal reduct is a reduct with minimum cardinality. Finding a minimal reduct is NP-hard [22], because all the possible subsets of conditional features must be generated to retrieve such a reduct. Therefore finding a near optimal reduct has generated much of interest [24,26,23]. Fig. 2 represents the steps of QUICKREDUCT algorithm [23], which searches for a minimal subset without exhaustively generating all possible subsets.

3.5. Rough set extensions

Efforts have been made to connect the attribute reduction concept in rough sets theory to feature selection in machine learning and classification tasks. However, traditional rough set based attribute reduction (RSAR) has three shortcomings which make it ineffective in real world applications [22,36,26]. Firstly, it only operates effectively with datasets containing discrete values and therefore it is necessary to perform a discretization step for real-valued attributes, secondly, RSAR is highly sensitive to noisy data and finally, RSAR methods examine only the information contained within the lower approximation of a set ignoring the information contained in the boundary region. Several extensions to the original theory have been proposed to overcome such shortcomings. Three well known extensions are variable precision rough sets (VPRS) [58], tolerance rough set model (TRSM) [44], and fuzzy rough sets (FRS) [22,13].

VPRS [58] attempts to overcome traditional rough sets shortcomings by generalizing the standard set inclusion relation $(\subseteq)$. In the generalized inclusion relation, a set X is considered to be a subset of Y if the proportion of elements in X which are not in Y is less than a predefined threshold. However, the introduction of a suitable threshold requires more information than contained within the data itself. This is contrary to the rough sets theory and OSF consideration of operating with no domain knowledge.

TRSM [44] uses a similarity relation instead of indiscernibility relation to relax the crisp manner of classical rough sets theory. As equivalence classes (elementary sets) in classical rough sets, tolerance classes are generated using similarity relation in TRSM, which are used to define lower and upper approximations. TRSM has two deficiencies which are contrary two our OSF considerations; First, generating tolerance classes needs a tolerance threshold, which is human defined. Second, the time complexity of generating all tolerance classes, using attribute subset B, is $\Theta(|B||U|^2)$, which is equal to worst-case time complexity of the PARTITION algorithm.

FRS [22,13] uses fuzzy equivalence classes generated by a fuzzy similarity relation to represent vagueness in data. Fuzzy lower and upper approximations are generated based on fuzzy equivalence classes. These approximations are extended versions of their crisp notions in classical rough sets, except that in the fuzzy approximations, elements may have membership degree in the range [0, 1]. FRS needs no extra knowledge to define operations on a given dataset. However, generating fuzzy equivalence classes in FRS is an expensive routine ($\Theta(|B||U|^2)$).

THE VPRS descriptions. Let $X, Y \subseteq U$, the relative degree of misclassification of X with respect to Y is defined as

$$c(X,Y) = \begin{cases} 1 - \frac{|X \cap Y|}{|U|} & \text{if } |X| > 0 \\ 0 & \text{if } |X| = 0 \end{cases} \tag{8}$$

Based on definition of relative misclassification, the generalized inclusion relation is defined as

$$X \overset{\beta}{\subseteq} Y \iff c(X,Y) \leq \beta, \quad 0 \leq \beta \leq 0.5 \tag{9}$$

where β is predefined level of error.

By replacing the inclusion relation with generalized inclusion relation, the B-lower and B-upper approximations of set X can now be redefined as

$$\underline{B}_\beta X = \{x | [x]_B \overset{\beta}{\subseteq} X\} \tag{10}$$

$$\overline{B}_\beta X = \{x | [x]_B \overset{\beta}{\subseteq} X\} \tag{11}$$

In addition to rough sets extensions, there are also some modifications, which do not change classical rough sets principles. The dependency notion in classical rough sets is redefined in [36] and [14] to deal with useful information that may be contained in the boundary region. Unlike the extensions above, these modifications do not redefine the lower and upper approximations in classical rough sets and therefore need no human input knowledge to deal with available data. Moreover the PARTITION algorithm may be applied directly; that is, generating equivalence classes in the modifications is more efficient than generating tolerance classes and fuzzy equivalence classes in TRSM and FRS, respectively. Since the modification in [14] has been used as a part of our work, we review more extensively in the following subsection.

3.6. Useful information in boundary region and the noise resistant dependency measure

Almost all the classical rough set-based attribute reduction methods use only the information contained in the positive region. However the boundary region may also contain useful information which is ignored [36,14]. Such a scenario is common in real-valued datasets, where some adjacent values may be placed in different regions because of the crisp manner of classical rough sets. Measuring the information contained in the boundary region and using this value combined with the classical rough set-based dependency value could provide a better qualification of the existing dependency between feature subsets. The method proposed in [14] uses a combination of the proximity measure proposed in [36] and a noise resistant measure to qualify the information contained in the boundary region.

The proximity measure proposed in [36] uses a distance metric to calculate the proximity of the boundary and positive regions. Let X be a set of objects and B a subset of attributes. The mean positive region, which is the mean of all object attribute values in $POS_B(X)$, is defined as

$$m = \left\{ \frac{\sum_{x \in \underline{B}X} f(x)}{|POS_B(X)|} : \forall f \in B \right\} \tag{12}$$

The proximity of any object $y \in BND_B(X)$ from the mean positive region is defined as

$$\delta_B(m,y) = \begin{cases} d(m,y) & \text{if } |POS_B(X)| \neq 0 \\ 0 & \text{if } |POS_B(X)| = 0, \end{cases} \tag{13}$$

where d can be any distance function such as Euclidean distance metric.

The proximity of the boundary region to the positive region is defined as

$$\omega(C,D) = \begin{cases} \psi_B^{-1} & \text{if } |BND_C(D)| \neq 0 \\ 1 & \text{if } |BND_C(D)| = 0 \end{cases} \tag{14}$$

where

$$\psi_B = \sum_{y \in BND_B(X)} \delta_B(m,y) \tag{15}$$

Table 2
An example of noisy data.

$x \in U$	a	b
1	0	0
2	0	0
3	0	0
4	0	1
5	1	1
6	1	1
7	1	1
8	1	0

Although the proximity measure in (14) enables the attribute reduction algorithm to deal with real-valued datasets, this measure is insufficient to control noisy data, specially in non-real-valued datasets. For example suppose two binary attributes a and b (Table 2), where a is 0 for all first half instances and 1 for all second half and b is similar to a, except that a value from the first half is exchanged with a value from second half. Obviously b is a noisy version of a. Let $C = \{a\}$ and $D = \{b\}$ then

$$U/C = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}\}$$

and

$$U/D = \{\{1, 2, 3, 8\}, \{4, 5, 6, 7\}\}.$$

Hence

$$POS_C(D) = \bigcup \left\{\underline{C}\{1, 2, 3, 8\}, \underline{C}\{4, 5, 6, 7\}\right\} = \emptyset$$

Therefore $\gamma(C, D) = 0$ based on (7), which implies that the two attribute sets are independent from the classical rough sets viewpoint. Moreover they have no dependency even if we use a combination of (7) and (14), because $\omega(C, D) = 0$.

The problem comes from the crisp manner of the inclusion relation in defining lower approximations. Consider for example the case $\underline{C}\{1, 2, 3, 8\}$. This approximation is an empty set, because none of the elementary sets in U/C are a subset of $\{1, 2, 3, 8\}$. However consider the elementary set $\{1, 2, 3, 4\}$ in U/C. If we remove 4 (the noise element) from this subset, the positive region will not be empty any more. The same rule can be applied to $\{5, 6, 7, 8\}$, removing 8 (another noise element).

The approach described here uses a noise resistant dependency measure to search for reducts. This measure uses the information that may be hidden due to noise elements and assigns a significance value to this information.

3.6.1. Impurity rate and noise resistant measure

The noise resistant measure attempts to qualify the information that may be unseen due to the crisp manner of the inclusion relation in defining lower approximations. This measure uses an impurity rate value to calculate the noisy portion of a set. Let A and B be two sets. The impurity rate of A with respect to B can be defined as follows:

$$c(A, B) = \frac{|A - B|}{|A|} \tag{16}$$

This value calculates the portion of the elements that must be eliminated from A to make it totally included in B. It is important to note that if $c(A, B) > 0.5$, the impurity of A with respect to B is more than its impurity with respect to $\overline{B}$. In this case, A may be supposed to be a noisy version of $\overline{B}$ and all elements in $A \cap B$ will constitute the noisy portion of A. Therefore, the B-related information that could be retrieved after removing impurities from A can be formulated as

$$\xi(A, B) = \begin{cases} 1 - c(A, B) & \text{if } c(A, B) \leq 0.5 \\ 0 & \text{if } c(A, B) > 0.5 \end{cases} \tag{17}$$

This formulation can be applied to elementary sets to extract information that may be unseen in calculating lower approximations. In this regard, a noise measure function, ϕ, is defined as

$$\phi_B(X) = \frac{\sum_{Y \in U/B} \xi(Y, X) \ \ [\xi(Y, X) \neq 1]}{|U/B|} \tag{18}$$

This function quantifies the possibility of transferring some objects from the boundary to the positive region of a set, if the noise elements could be removed.

Let C and D be two attribute sets. The noisy dependency of D on C can be defined as follow:

$$\nu(C, D) = \sum_{Y \in U/D} \phi_C(Y) \tag{19}$$

NRQUICKREDUCT(C, d)
C: The set of all conditional features
d: The decision feature

1: $R \leftarrow \{\}$
2: **while** $(\gamma(R, d) \neq \gamma(C, d))$ **do**
3: $\quad x \leftarrow \arg\max_{f \in C-R} (\rho(R \cup \{f\}, d) - \rho(R, d))$
4: $\quad R \leftarrow R \cup \{x\}$
5: **end while**
6: **return** R

Fig. 3. The noise resistant-assisted QUICKREDUCT algorithm.

Consider the dataset in Table 2. The noisy dependency of $D = \{b\}$ on $C = \{a\}$ is

$$\begin{aligned} \nu(C, D) &= \phi_C(\{1, 2, 3, 8\}) + \phi_C(\{4, 5, 6, 7\}) \\ &= \frac{\xi(\{1, 2, 3, 4\}, \{1, 2, 3, 8\}) + \xi(\{5, 6, 7, 8\}, \{1, 2, 3, 8\})}{2} \\ &+ \frac{\xi(\{1, 2, 3, 4\}, \{4, 5, 6, 7\}) + \xi(\{5, 6, 7, 8\}, \{4, 5, 6, 7\})}{2} \\ &= \frac{\frac{3}{4} + 0}{2} + \frac{0 + \frac{3}{4}}{2} = 0.75 \end{aligned}$$

Such as the proximity measure in (14), the noisy dependency operates on boundary region. However the proximity measure considers each point in the boundary region separately and calculates its distance from the positive region, while the noisy dependency considers subsets of objects to measure their transmission possibility to the positive region. Therefore the two values are combined to create a new measure for evaluating boundary region as

$$\tau(C, D) = \omega(C, D) + \nu(C, D) \tag{20}$$

This new measure can be used alongside the classical dependency. As one measure only operates on the objects in boundary region and the other only on the objects in positive region, the two operators may be combined to create a noise resistant evaluation measure ρ:

$$\rho(C, D) = \frac{\tau(C, D) + \gamma(C, D)}{2} \tag{21}$$

3.6.2. Noise resistant-aided rough set attribute reduction (NRRSAR)

The proposed noise resistant measure can be used as an attribute selection criterion in rough set based attribute reduction algorithms. Fig. 3 displays the NRQUICKREDUCT algorithm, which is similar to QUICKREDUCT, but uses the proposed measure to guide the attribute reduction process.

The algorithm starts with an empty selected attribute set R. At each iteration, an attribute is added to R from non-selected attributes. This attribute is selected such that its addition causes maximum increase in the value of the noise resistant measure. This process continues until the dependency value of the selected attributes $(\gamma(R, d))$ equals the dependency of complete conditional attribute set $(\gamma(C, d))$. It is important to point out that the new measure is used as an attribute selection criterion, not as a termination criterion.

4. Online streaming feature selection using NRRSAR

In this section, we first define online streaming features and then propose a new algorithm to implement the NRRSAR for feature selection with streaming features.

Suppose that $DS_t = (A_t, F_t, d)$ is a decision system at time t where $A_t = \{x_{1_t}, x_{2_t}, \ldots, x_{N_t}\}$, $F_t = \{f_{1_t}, f_{2_t}, \ldots, f_{M_t}\}$ and d is a decision feature. In online streaming features (OSF), new conditional features flow in one by one over time while the number of objects in A remains fixed. In other words, for every time $t' > t$, $M_{t'} \geq M_t$ while $N_{t'} = N_t$.

Because we do not have access to the full feature space in the online streaming features context, we need to gradually build a reduct over time based on features seen so far. A batch version of NRRSAR, such as NRQUICKREDUCT, is not directly applicable to generate a reduct in OSF scenarios. The problem is that the feature selection criterion is defined so that the algorithm needs to access the full non-selected feature space in order to find the next feature. This problem can be relaxed by allowing a new incoming feature to be included in the selected subset, if it increases the noise resistant measure. However, this relaxation may be dangerous in OSF scenarios, because early incoming features will be selected with more chance. For example suppose a large dataset with 100 000 features, which we need to consider in streamwise manner. It is possible that the dataset becomes consistent (γ(selected subset, $d) = 1$) using some (say 10) early streamed features and therefore, the algorithm will not consider the other remaining 99 990 features.

OS-NRRSAR-SA(d)
d: The decision feature

```
1: R ← {},
2: do
3:     f ← GET_NEW_FEATURE()
4:     if (γ(R, d) ≠ 1)
5:         if (γ(R ∪ {f}, d) − γ(R) > 0) or (ρ({f}, d) > 0))
6:             R ← R ∪ {f}
7:         end if
8:     else
9:         R1 ← R ∪ {f}
10:        B ←NON-SIGNIFICANT(R1, f)
11:        if (|B| > 1)
12:            R ← R1 − B
13:        end if
14:        if (|B| = 1)
15:            ρ1 = ρ(f, d)
16:            ρ2 = ρ(f′, d), f′ ∈ B
17:            if (ρ1 > ρ2)
18:                x ← f′
19:            else if (ρ1 < ρ2)
20:                x ← f
21:            else
22:                x ← RANDOM{f, f′}
23:            end if
24:            R ← R1 − x
25:        end if
26:    end if
27: until (stopping criterion is met)
```

Fig. 4. The online streaming NRRSAR using significance analysis.

4.1. Online streaming NRRSAR using significance analysis

Significance analysis is a tool for measuring the effect of removing an attribute, or a subset of attributes, from a decision system on the positive region defined by that decision system. Let $DS = (A, F, d)$ be a decision system. The significance of attribute $f \in F$, denoted $\sigma_{(F,d)}(f)$, is defined as [45]

$$\sigma_{(F,d)}(f) = \frac{\gamma(F,d) - \gamma(F - \{f\}, d)}{\gamma(F,d)} \tag{22}$$

Moreover, the significance of attribute set $C \subseteq F$, denoted $\sigma_{(F,d)}(C)$, is defined as

$$\sigma_{(F,d)}(C) = \frac{\gamma(F,d) - \gamma(F - C, d)}{\gamma(F,d)} \tag{23}$$

Fig. 4 represents the online streaming version of NRRSAR which uses significance analysis to control the inclusion of any new incoming feature in the selected feature set. The algorithm starts with an empty selected subset R. Then it waits for a new incoming feature (line 3). Once a new feature f is provided, the algorithm proceeds as follows:

(a) If the dataset is not consistent, the algorithm calculates two values; 1) the increase of the dependency value, when f is added to current subset, and 2) the noise resistant dependency of d on f (line 5). If at least one of these values is no-zero, the current subset is updated to include f (line 6), otherwise, f is rejected.
(b) If the dataset is consistent, the new incoming feature is not eliminated immediately, but the algorithm checks to see if there exists any current reduct subset, which becomes non-significant due to the presence of f (lines 9–10). If such subset exists, and its size is larger than one, then the subset can be replaced with f (lines 11–13). This makes our dataset smaller, while keeping the consistency. Moreover, if only one feature (say f') becomes non-significant due to f, then one of the features f and f' is removed based on the noise resistant measure value. If the two features have the same noise resistant measure value, then the algorithm removes one randomly. Finally if the new incoming feature makes no other feature non-significant, then the feature is ignored and the current selected subset remains unchanged.

The algorithm alternates the above two phases till the stopping criteria is satisfied. Different stopping criterions can be adopted to control the algorithm execution. If the size of the streaming dataset is known, the algorithm can keep running until the last feature (no further features are available). However if we have no knowledge about the feature space (including maximum number of features), then the algorithm can stop once a predefined accuracy is satisfied or a maximum number of iterations is reached.

4.1.1. Consistency and selected subset changes

Let $F_t = \{f_1, f_2, \ldots, f_t\}$ be the set of features that have streamed in until the time t, such that f_i arrives before f_j if and only if $i < j$. The following theorems help us to have a better insight into the selected subset growth and the consistency of the dataset from the beginning to the time t.

Theorem 1. *(See [25].) Suppose that R is a subset of the set of conditional features, x is an arbitrary conditional attribute that belongs to the dataset, and d is the set of decision attributes that dataset objects are classified into. Then $\gamma(R \cup \{x\}, d) \geq \gamma(R, d)$.*

Theorem 2. *Let R_t be the selected feature subset using OS-NRRSAR-SA at time t. IF $\gamma(R_t, d) = 1$, then $\forall t' \geq t, \gamma(R_{t'}, d) = 1$ and $|R_{t'}| \leq |R_t|$.*

Proof. We prove this theorem by induction. For $t' = t$, the theorem holds by assumption (the base case). Let $\gamma(R_{t'=t+k}, d) = 1$ for a given $k \geq 1$. Suppose that a new feature f_{t+k+1} is streamed in at time $t' = t + k + 1$. Then the second phase of the algorithm will be triggered, because the decision system is consistent using current selected feature subset R_{t+k}. In this phase a non-significant feature subset B will be found by the NON-SIGNIFICANT function. Two cases can occur after this, 1) f_{t+k+1} will be denied and $R_{t+k+1} = R_{t+k}$, and 2) B will be replaced with f_{t+k+1} and therefore $R_{t+k+1} = (R_{t+k} \cup f_{t+k+1}) - B$. In the former case, it is obvious that $\gamma(R_{t+k+1}, d) = 1$ and $|R_{t+k+1}| = |R_t + k|$. In the later case

$$\sigma_{(R_{t+k} \cup \{f_{t+k+1}\}, d)}(B) = \frac{\gamma(R_{t+k} \cup \{f_{t+k+1}\}, d) - \gamma((R_{t+k} \cup \{f_{t+k+1}\}) - B, d)}{\gamma(R_{t+k} \cup \{f_{t+k+1}\}, d)} = 0$$

then

$$\gamma((R_{t+k} \cup \{f_{t+k+1}\}) - B, d) = \gamma(R_{t+k} \cup \{f_{t+k+1}\}, d)$$

According to monotonicity of γ, $\gamma(R_{t+k} \cup \{f_{t+k+1}\}, d) = 1$, and therefore $\gamma(R_{t+k+1}, d) = \gamma((R_{t+k} \cup \{f_{t+k+1}\}) - B, d) = 1$. Moreover, $|R_{t+k+1}| \leq |R_{t+k}|$, because $|B| \geq 1$. Therefore the prove of the induction step is complete. □

Theorem 3. *Let R_t be the selected feature subset using OS-NRRSAR-SA at time t. IF $\gamma(R_t, d) < 1$, then $\forall t' < t, \gamma(R_{t'}, d) \leq \gamma(R_t, d)$ and $|R_{t'}| \leq |R_t|$.*

Proof. On the contrary, suppose that $\exists t' < t$, such that $|R_{t'}| > |R_t|$ or $\gamma(R_{t'}, d) > \gamma(R_t, d)$. If $|R_{t'}| > |R_t|$ then it is obvious that $R_{t'} \not\subseteq R_t$. On the other hand, if $\gamma(R_{t'}, d) > \gamma(R_t, d)$, we can conclude again that $R_{t'} \not\subseteq R_t$ by Theorem 1. Then in either case, $\exists a \in R_{t'}$ s.t. $a \notin R_t$ and hence $\exists t_1, t \leq t_1 < t'$, when a is removed from selected subset. Removing feature(s) from selected subset only occurs during the second phase of the algorithm and this phase triggers if the decision system is consistent using selected subset. Therefore $\gamma(R_{t_1}, d) = 1$. However, by Theorem 2, we have $\gamma(R_t, d) = 1$, which is a contradiction. □

Theorems 2 and 3 enable us to divide F_t into two disjoint feature subsets $F_1 = \{f_1, f_2, \ldots, f_i\}$ and $F_2 = \{f_{i+1}, f_{i+2}, \ldots, f_t\}$, such that, when the features in F_1 are present, the dataset is not consistent, while it is consistent when the features in F_2 are present. Moreover, based on these theorems, the size of the selected subsets constitutes a sequence over time. Starting from the first element (the size of the first subset), we encounter elements in non-decreasing order until we reach the maximum element in the list, after which we encounter elements in non-increasing order. The maximum element in the list corresponds to the time that the last element in the F_1 is arrived.

4.1.2. The NON-SIGNIFICANT function in OS-NRRSAR-SA

In Fig. 4, the algorithm uses the notation NON-SIGNIFICANT(R_1, f) routine to identify features, which became non-significant in R_1 due to the feature f. Several implementations of the routine can be adopted based on the relative importance of the reduct size compared with the time required to locate the non-significant features. If the goal is to keep the reduct as small as possible, then an implementation which finds the largest non-significant subset can be adopted, i.e.:

$$B \leftarrow \arg\max_{|S|} \left(\sigma_{(R1,d)}(S) = 0, \quad S \subseteq R_1 - \{f\}\right) \tag{24}$$

However, this implementation needs to consider all possible non-significant subsets of the current reduct, which itself needs the total number of the current reduct subsets to be checked. It is clear that this number is exponentially growing with the

```
NON-SIGNIFICANT(R, f)
R: The feature set
f: The newly added feature

1: B ← {},
2: T ← R − {f},
3: while (|T| ≠ 0) do
4:     g ← RANDOM{f_i ∈ T, i = 1, 2, ..., |T|}
5:     if (σ_(R,d)(g) = 0)
6:         B ← B ∪ {g}
7:         R ← R − {g}
8:     end if
9:     T ← T − {g}
10: end while
11: return B
```

Fig. 5. An implementation of the NON-SIGNIFICANT function which uses a sequential backward elimination of non-significant features.

size of the reduct. On the other hand, we can use a sequential backward elimination method represented in Fig. 5, if we need a very time-efficient reduct updates. Starting from full reduct, at each step the method considers a random feature that has not already been evaluated and drops the feature out if it is non-significant based on available feature subset. Because of random consideration of features, different executions of this method may return different non-significant subsets and therefore a good heuristic would be to executing the method k times and selecting the result with maximum size.

4.1.3. The time complexity of OS-NRRSAR-SA

The time complexity of OS-NRRSAR-SA depends on the number of tests. Two types of tests are used in the algorithm; the γ-tests and the ρ-tests. As stated previously, the time required by this RS-based tests can be attributed by the time that is required to generate equivalence classes (the PARTITION algorithm).

Suppose that at time t a new feature f_t be present to the OS-NRRSAR-SA algorithm and let R_t be the selected feature subset at this time. If the available dataset is not consistent using R_t, the first phase of the algorithm will be triggered. This phase includes a single ρ-test and therefore, the worst-case time complexity of this phase is $\Theta(|R_t||U|^2)$. However, if the dataset is consistent, the second phase will be triggered, which needs several γ-tests to remove non-significant features. The time complexity of this phase depends on the adopted implementation of the NON-SIGNIFICANT function. Using the implementation (24), the algorithm needs $2^{|R_t|}$ tests and therefore the worst-case time complexity of the phase will be $O(2^{|R_t|}|R_t||U|^2)$. However if the algorithm adopts the method represented in Fig. 5, the wort-case time complexity of the phase will be $O(|R_t|^2|U|^2)$.

Let $F = \{f_1, f_2, \ldots, f_M\}$ is the set of features that have arrived so far. Assuming that the dataset be non-consistent for the $|M_1|$ first incoming features and consistent for the remaining $|M_2|$ features, where $|M_1| + |M_2| = |M|$. The selected subset with maximum size is located after arriving f_{M_1} and therefore, the worst-case time complexity of the algorithm, for the implementation (24) of the NON-SIGNIFICANT function, is

$$O\left((|M_1||R_{M_1}||U|^2) + (|M_2|2^{|R_{M_1}|}|R_{M_1}||U|^2)\right)$$

and for the implementation represented in Fig. 5, is

$$O\left((|M_1||R_{M_1}||U|^2) + (|M_2||R_{M_1}|^2|U|^2)\right)$$

It is clear that the time complexity of OS-NRRSAR-SA is independent of the total number of features. Although the worst case time complexity of the proposed algorithm is square with respect to the number of selected features, in many real-world applications, only a small number of features in a large feature space are predictive and relevant to decision feature [53]. Therefore $|R_t|$ (and hence $|R_{M_1}|$) is so small that its square does not affect the time complexity of the OS-NRRSAR-SA algorithm, significantly.

Being squared with respect to $|U|$ (number of training instances) is because of the fact that we considered the worst-case time complexity of the PARTITION algorithm, for analyzing the complexity of the proposed algorithm. As stated in section 3.1, the time complexity of the PARTITION algorithm for calculating U/B is $\Theta(|B||P||U|)$, where $|P|$ is the number of generated B-elementary subsets. The worst-case time complexity of the PARTITION algorithm, which is $O(|B||U|^2)$, occurs when none of the objects in U are indiscernible according to B, i.e. $|P| = |U|$. In order to investigate $|P|$ from an application viewpoint, 10 artificial datasets, organized in two groups, were generated. The first group were generated to investigate the effect(s) of the number of features and instances on the number of generated partitions. In this regard, five datasets each with 30 uniformly distributed binary features were generated randomly. The five datasets in this group were different in terms of the number of instances. The second group, on the other hand, were generated to investigate the effects of data

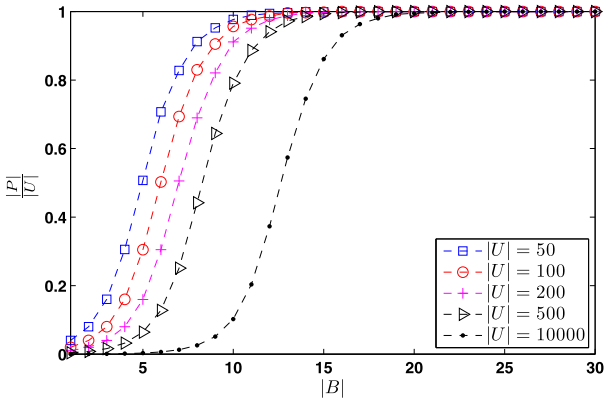

Fig. 6. The effects of $|B|$ and $|U|$ on $|P|$.

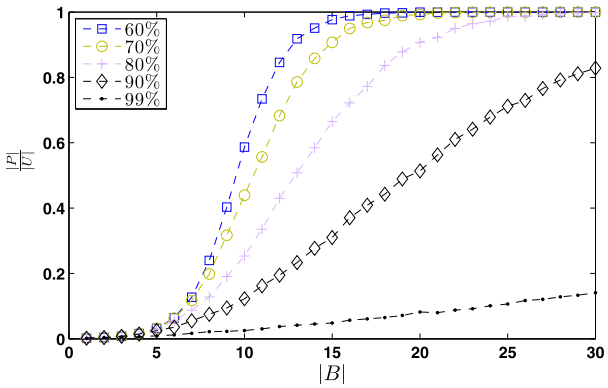

Fig. 7. The effects of sparseness on $|P|$.

sparseness (bias of feature values to a special value) on $|P|$. The five datasets in this group were all binary with 30 features and 1000 instances, but different in terms of sparseness (sparseness of 50% means uniformly distributed feature values).

For each dataset, the PARTITION algorithm was applied with feature subsets of sizes ranging from 1 to 30. For each subset size, several different combinations of features were selected randomly and the average ratio of the generated partitions with respect to the number of instances ($|P|/|U|$) was recorded. Fig. 6 represents the results for the first group. The figure shows the fast decreasing of the ratio $|P|/|U|$, once $|B|$ becomes smaller than a threshold. The threshold is different for each dataset and the larger the $|U|$ the larger the threshold. Fig. 7 represents the results for the second group. As it can be seen from this figure, the more sparse the dataset, the higher the possibility of the objects become indiscernible and therefore the ratio $|P|/|U|$ is significantly small, even for large values of $|B|$.

As stated, the proposed OS-NRRSAR-SA algorithm keeps the $|R_t|$ (number of selected features at time t) very small and therefore all the PARTITION calls (in γ and ρ tests) will be executed with small feature subsets. Therefore $|P| \ll |U|$ and the PARTITION algorithm will be very time efficient. Moreover, most of the real world large scale datasets are highly sparse, which means even faster executions of the PARTITION calls.

4.1.4. A worked example

To illustrate the operation of the new proposed algorithm, a small example dataset is considered, containing discrete-valued conditional and decision attributes. Crisp data are used in this example to aid explanation of the approach. Table 3 contains ten objects. It has 5 crisp-valued conditional attributes and a single crisp-valued decision attribute. The streaming order of conditional features is the same as their order in the table. The algorithm starts with an empty selected subset R, then waits for a new incoming feature. In the following we describe steps of the algorithm for each streaming feature. The calculating details of the dependency measures (γ and ρ) are represented only for the first feature (f_1) while for the remaining features (f_2–f_5) we have used the final pre-calculated values.

Once f_1 is streamed in, the first phase of the algorithm will be triggered because $\gamma(R, d) = 0 < 1$. This phase tests the following condition and adds f_1 to R if this condition holds.

$$(\rho(\{f_1\}, d) > 0) \text{ or } (\gamma(\{f_1\}, d) > 0)$$

Table 3
An example dataset.

$x \in U$	f_1	f_2	f_3	f_4	f_5	d
1	1	0	0	1	0	0
2	2	0	2	0	2	4
3	1	1	0	0	0	0
4	1	2	1	2	2	3
5	2	2	0	0	2	2
6	0	0	1	2	2	3
7	2	1	0	1	0	0
8	2	1	1	2	1	2
9	1	0	2	0	0	2
10	1	1	2	1	2	4

Based on equations (20) and (21)

$$\rho(\{f_1\}, d) = \frac{\nu(\{f_1\}, d) + \omega(\{f_1\}, d) + \gamma(\{f_1\}, d)}{2}$$

using equation (7), $\gamma(f_1, d)$ can be calculated:

$$\begin{aligned}\gamma(\{f_1\}, d) &= \frac{|POS_{\{f_1\}}(d)|}{10} = \frac{|\bigcup_{X \in U/d} \underline{\{f_1\}} X|}{10} \\ &= \frac{|\underline{\{f_1\}}(\{1,3,7\}) \cup \underline{\{f_1\}}(\{5,8,9\}) \cup \underline{\{f_1\}}(\{2,10\}) \cup \underline{\{f_1\}}(\{4,6\})|}{10} \\ &= \frac{|\emptyset \cup \emptyset \cup \emptyset \cup \{6\}|}{10} = 0.1\end{aligned}$$

It is worth noting at this point, that having calculated $\gamma(\{f_1\}, d)$, we can conclude that f_1 must be added to R. However, in order to keep the generality, we continue with calculating $\omega(\{f_1\}, d)$ and $\nu(\{f_1\}, d)$.

To calculate $\omega(\{f_1\}, d)$, positive region mean must first be calculated. As calculated above, $POS_{\{f_1\}}(d) = \{6\}$ and therefore based on equation (12)

$$m = f_1(x = 6) = 0$$

The distance of any object in the boundary region from the mean positive region can now be calculated. As mentioned in section 3.6, there are many distance metrics which can be applied. In the approach documented here, a variation of Euclidean distance is used. This distance, which is proposed in [36], is defined as

$$\delta_P(m, y) = \sqrt{\sum_{a \in P} f_a(m, y)^2}, \tag{25}$$

where

$$f_a(m, y) = \begin{cases} 1 & \text{if } a(m) \neq a(y) \\ 0 & \text{if } a(m) = a(y) \end{cases}$$

From this, the distances of all of the objects in the boundary region to m can now be calculated. The individual distances of objects in $BND_{\{f_1\}}(d)$ can be shown to be

$$object1 : \sqrt{f_{f_1}(6,1)} = 1$$

$$object2 : \sqrt{f_{f_1}(6,2)} = 1$$

$$object3 : \sqrt{f_{f_1}(6,3)} = 1$$

$$object4 : \sqrt{f_{f_1}(6,4)} = 1$$

$$object5 : \sqrt{f_{f_1}(6,5)} = 1$$

$$object7 : \sqrt{f_{f_1}(6,7)} = 1$$

$$object8 : \sqrt{f_{f_1}(6,8)} = 1$$

$$object9 : \sqrt{f_{f_1}(6,9)} = 1$$

$$object10 : \sqrt{f_{f_1}(6,10)} = 1$$

Thus from equation (14):

$$\omega(\{f_1\}, d) = (1+1+1+1+1+1+1+1+1+1)^{-1} = \frac{1}{9}$$

The noisy dependency of d on f_1 can be calculated using equation (19)

$$\begin{aligned}
\nu(\{f_1\}, d) &= \sum_{Y \in U/d} \phi_{\{f_1\}}(Y) \\
&= \phi_{\{f_1\}}(\{1,3,7\}) + \phi_{\{f_1\}}(\{5,8,9\}) + \phi_{\{f_1\}}(\{2,10\}) + \phi_{\{f_1\}}(\{4,6\}) \\
&= \frac{\xi(\{1,3,4,9,10\}, \{1,3,7\}) + \xi(\{2,5,7,8\}, \{1,3,7\}) + \xi(\{6\}, \{1,3,7\})}{3} \\
&+ \frac{\xi(\{1,3,4,9,10\}, \{5,8,9\}) + \xi(\{2,5,7,8\}, \{5,8,9\}) + \xi(\{6\}, \{5,8,9\})}{3} \\
&+ \frac{\xi(\{1,3,4,9,10\}, \{2,10\}) + \xi(\{2,5,7,8\}, \{2,10\}) + \xi(\{6\}, \{2,10\})}{3} \\
&+ \frac{\xi(\{1,3,4,9,10\}, \{4,6\}) + \xi(\{2,5,7,8\}, \{4,6\}) + \xi(\{6\}, \{4,6\})}{3} \\
&= \frac{0+0+0.5}{3} = \frac{1}{6}
\end{aligned}$$

Having calculated the three measures, the noisy dependency measure can be calculated:

$$\rho(\{f_1\}, d) = \frac{\frac{1}{10} + \frac{1}{9} + \frac{1}{6}}{2} = 0.1889 > 0$$

and therefore f_1 must be added to current selected features, i.e. $R = \{f_1\}$.

Considering f_2, the dependency, proximity, and noisy measures can be shown to be 0, 0.1, and 0.5, respectively and hence $\rho(\{f_2\}, d) = 0.3 > 0$. Therefore f_2 must be added to R.

At the incoming time of f_3, $\gamma(R, d) = \gamma(\{f_1, f_2\}, d) = 0.4 \neq 1$, and therefore the first phase of the algorithm triggers again. This feature will be added to R, because $\gamma(\{f_3\}, d) = 0$, $\omega(\{f_3\}, d) = 0.1$, and $\nu(\{f_3\}, d) = 0.6944$ and therefore $\rho(\{f_3\}, d) = 0.3972$.

When f_4 streams in, $\gamma(R, d) = \gamma(\{f_1, f_2, f_3\}, d) = 1$, which means that the decision system is consistent using current selected subset and therefore, the second phase of the algorithm triggers. In this phase the algorithm checks to see if there exists any current reduct subset, which becomes non-significant due to the presence of new incoming feature. This is done using the NON-SIGNIFICANT function. If we use the implementation (24) of this function then

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1\}) = 0.2$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_2\}) = 0$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_3\}) = 0$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1, f_2\}) = 0.7$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1, f_3\}) = 0.4$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_2, f_3\}) = 0.6$$

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1, f_2, f_3\}) = 1$$

From this it follows that one of the attributes f_2 and f_3 (not together) is non-significant. Considering $B = \{f_2\}$, f_4 will be preserved while f_2 will be removed from R. This is because of the fact that $|B| = 1$ and $\rho(\{f_4\}, d) = 0.3556 > \rho(\{f_2\}, d) = 0.3$. On the other hand, considering $B = \{f_3\}$, f_4 will be removed, because $\rho(\{f_4\}, d) = 0.3556 < \rho(\{f_3\}, d) = 0.3972$. For the remainder of this example, the former consideration is applied and hence $R = \{f_1, f_3, f_4\}$.

Considering f_5, the second phase is triggered again. The significance of subsets in R can be shown to be

$$\sigma_{(\{f_1,f_3,f_4,f_5\},d)}(\{f_1\}) = 0$$

$$\sigma_{(\{f_1,f_3,f_4,f_5\},d)}(\{f_3\}) = 0.4$$

$$\sigma_{(\{f_1,f_3,f_4,f_5\},d)}(\{f_4\}) = 0$$

Table 4
Summary of the benchmark high dimensional datasets.

Dataset	# Attributes	# Train	# Test	Type
dorothea	100 000	800	800	C
arcene	10 000	100	700	I
dexter	20 000	300	2000	I
madelon	500	2000	1800	C
sido0	4932	12 678	10 000	C
cina0	132	16 033	10 000	I, C
nova	16 969	1754	17 537	C
sylva	216	13 086	130 854	I, C
hiva	1617	3845	38 449	C
arrhythmia	279	452	–	C, I, R
mf	649	2000	–	I, R
tm1	100	1000	1000	I
tm2	100	1000	1000	I
tm3	1000	100	1000	I

C: categorical, R: real, I: integer.

$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1,f_3\}) = 0.4$$
$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1,f_4\}) = 0$$
$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_3,f_4\}) = 0.7$$
$$\sigma_{(\{f_1,f_2,f_3,f_4\},d)}(\{f_1,f_2,f_3\}) = 0.9$$

From this it follows that $B = \{f_1, f_4\}$. Because the size of B is larger than one, this subset can be replaced with f_4, and therefore the final selected subset is $R = \{f_3, f_5\}$.

5. Experimental results

In this section, we provide several experimental results to illustrate the performance of the proposed method. To do this, we compare the performance of the proposed OSFS algorithm (OS-NRRSAR-SA) with three existing OSFS algorithms, grafting, information-investing and fast-OSFS. Table 4 summarizes the 14 high dimensional datasets used in our experiments. In order to provide OSF scenario, features are considered one by one. The *dorothea*, *arcene*, *dexter*, and *madelon* datasets are from the NIPS 2003 feature selection challenge [7]. The *nova*, *sylva*, and *hiva* datasets are from WCCI 2006 Performance Prediction Challenges [8]. The *sido0* and *cina0* datasets are from the WCCI 2008 Causation and Prediction Challenges [9]. The *arrhythmia* and *mf* (multiple features) datasets are selected from the UCI machine learning repository [2], and the *tm1*, *tm2*, and *tm3*, three synthetic problems called threshold max, are from [39]. All the experiments are carried out on a DELL workstation with Windows 7, 2 GB memory, and 2.4 GHz CPU. Four classifiers are employed for the classification of the data, J48 [52,43], JRip [10], Naive Bayes [12], and kernel SVM with RBF kernel function [4].

5.1. Experiments on parameters of OS-NRRSAR-SA

Here we consider the effect of different implementations of the NON-SIGNIFICANT function on the performance of the OS-NRRSAR-SA algorithm. The NON-SIGNIFICANT function tries to find those features, which become non-significant when a new feature is present to the current decision system. Three implementations are considered here; 1) The implementation related to (24), which finds the largest non-significant subset (max subset), 2) The sequential backward elimination method represented in Fig. 5 with $k = 1$ (SBE1), and 3) SBE with $k = 3$ (SBE3). The value of k represents the number of SBE executions for each NON-SIGNIFICANT function calls. In other words, for each non-significant call in OS-NRRSAR-SA algorithm, the SBE function is executed k times, each time with a different random consideration of features, and then a result with the maximum number of non-significant features is selected.

Fig. 8 reports the results of the three implementations in terms of selected subsets size, run-times, and classification accuracies. In this figure, we can see that the compactness of the max subset and SBE3 implementations are superior for most of the datasets, whereas the run-time curves in this figure show that the max subset implementation needs a considerable run-time in comparison with the other two implementations. Moreover, the SBE1 and SBE3 implementations are comparable in terms of run-times. For the *sido0* dataset, the SBE3 is faster than SBE1, but this relates to the fact that during the feature selection process, the SBE3 keeps the subset size very small in comparison with the SBE1 for this particular dataset. Comparing the three implementations in terms of classification accuracies demonstrates that the SBE3 implementation is superior for most of the datasets, using both classifies. Although the max subset classification accuracies are comparable with those obtained using SBE3, the very faster run-times of SBE3, makes it a preferable method for large scale data sets.

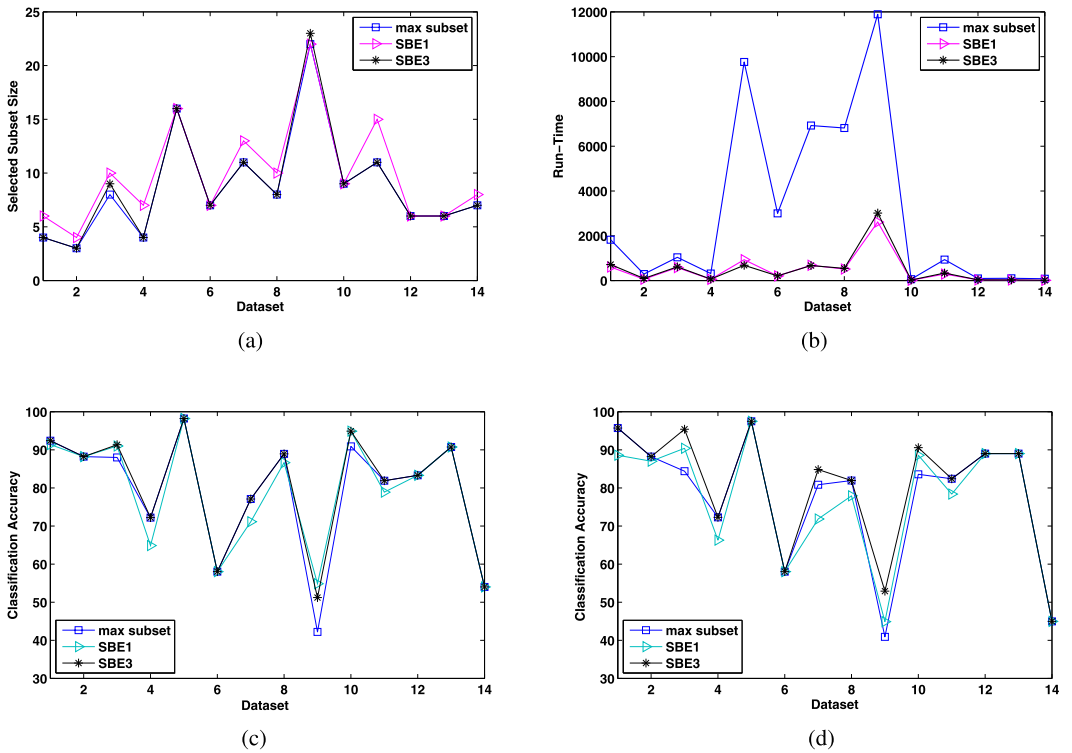

Fig. 8. Results using OS-NRRSAR-SA with different implementations of the NON-SIGNIFICANT function. a) selected subsets sizes, b) run-times (in seconds), c) classification accuracies using J48, d) classification accuracies using JRip (the labels of the x-axis from 1 to 14 denote the datasets: 1: dorothea; 2: arcene; 3: dexter; 4: madelon; 5: sido0; 6: cina0; 7: nova; 8: sylva; 9: hiva; 10: arrhythmia; 11: mf; 12: tm1; 13: tm2; 14: tm3).

5.1.1. Comparison of OS-NRRSAR-SA with grafting, information-investing, and fast-OSFS

Here, the OS-NRRSAR-SA algorithm is compared with grafting [40], information-investing [48], and fast-OSFS [53]. Because of the special results using DIA-RED [50], the comparisons with this algorithm are illustrated in a separate subsection. For the OS-NRRSAR-SA algorithm, the SBE3 implementation of the NON-SIGNIFICANT function is adopted. In the grafting algorithm, the multi-layer perceptron (MLP) is adopted as learning model and the λ parameters are chosen using 5-fold cross-validation on each of the training datasets. In the information-investing algorithm, the parameters are set as their default settings, $W_0 = 0.5$ and $W_\triangle = 0.5$. For the fast-OSFS algorithm, the independence tests are G^2 tests for all fully categorical or integer datasets and Fisher's z-tests for all datasets containing real-valued features. For both tests, the statistical significance level (α) is set to be 0.05.

Results are presented in terms of the selected subset size (compactness), the time to locate the subset (run-time) in seconds, the classification accuracy at the end of the streaming, and the classification accuracy of selected subsets during features streaming (in seconds).

Because of the fact that we do not have access to the full feature space, the streaming order of the features affects the final results. Therefore, in order to strengthen the comparison, we generated 30 different random streaming orders for each dataset. Then the hypothesis paired t-test is carried out to compare the results of the feature selection algorithms on these streaming orders. Let A and B be two methods and d_A, d_B be the set of results obtained using methods A and B, respectively, for different streaming orders. We define the following two one-tailed t-tests:

$$t1: \begin{cases} H_0 : \mu_{d_A} = \mu_{d_B} \\ H_1 : \mu_{d_A} > \mu_{d_B} \end{cases} \tag{26}$$

$$t2: \begin{cases} H_0 : \mu_{d_A} = \mu_{d_B} \\ H_1 : \mu_{d_B} > \mu_{d_A} \end{cases} \tag{27}$$

where, μ_D is the population mean of set D.

Based on results of these tests, the variable t is defined as

$$t = \begin{cases} \uparrow & \text{if the null hypothesis } (H_0) \text{ in } t_1 \text{ is rejected} \\ \downarrow & \text{if the null hypothesis in } t_2 \text{ is rejected} \\ = & \text{if none of the null hypothesis in } t_1 \text{ and } t_2 \text{ is rejected} \end{cases} \tag{28}$$

Table 5
Comparison of selected subsets size for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**4**	**7**	**4.7**	101	180	138.9	↓	92	119	103.2	↓	8	13	8.93	↓
arcene	**3**	**6**	**3.9**	12	20	15.8	↓	13	18	15.2	↓	–	–	–	–
dexter	**7**	**11**	**8.4**	14	24	19.1	↓	17	23	19.8	↓	10	13	11.6	↓
madelon	**4**	**5**	**4.4**	8	13	10.2	↓	6	11	8.6	↓	4	6	4.8	↓
sido0	**13**	**18**	**15.3**	74	115	89.5	↓	207	288	248	↓	46	80	61.3	↓
cina0	**6**	**9**	**6.9**	25	36	30.2	↓	15	18	16.3	↓	6	9	7.3	=
nova	**9**	**12**	**9.9**	37	65	51.9	↓	23	41	32.7	↓	10	15	13.4	↓
sylva	**7**	**11**	**8.8**	25	37	31.4	↓	29	43	36.3	↓	23	27	24.7	↓
hiva	**18**	**25**	**20.4**	271	409	344.8	↓	342	514	434.1	↓	29	41	38.9	↓
arrhythmia	**7**	11	**8.5**	11	16	13.9	↓	9	11	9.5	↓	7	10	8.5	=
mf	**11**	**15**	**13.1**	15	23	18.7	↓	11	17	13.8	=	11	18	14.2	↓
tm1	**6**	**7**	6.6	6	7	6.6	=	6	8	6.5	=	7	9	8.1	↓
tm2	**6**	**7**	6.5	6	7	6.4	=	6	8	6.7	=	7	8	7.5	↓
tm3	7	9	7.9	6	8	7.4	↑	7	9	8.2	=	–	–	–	–

Table 6
Comparison of run-times for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**647.1**	**843.3**	**726.3**	14 439	18 150	16 224	↓	12 347	14 399	13 342	↓	894.7	1001.3	928.5	↓
arcene	**83.4**	**118.4**	**93.1**	372.6	528.2	429.6	↓	259.6	334.1	297.3	↓	–	–	–	–
dexter	**421.5**	**642.9**	**562.8**	934.6	2461.3	1822.2	↓	1567.2	2266.3	1927.4	↓	600.3	713.8	649.2	↓
madelon	**56.0**	**67.5**	**61.3**	97.2	141.3	128.4	↓	95.2	163.5	130.3	↓	75.4	94.8	86.2	↓
sido0	**602.3**	**761.4**	**654.6**	5291	9942	8134.6	↓	7379	14734	10 631	↓	3244.2	5432.6	4735.6	↓
cina0	160.0	226.8	183.1	559.3	868.8	750.4	↓	531.7	628.1	590.0	↓	68.3	108.1	88.7	↑
nova	438.3	689.9	**508.3**	2218.9	4181.4	3210.79	↓	1030.9	3112.3	1539.4	↓	377.5	618.1	509.4	=
sylva	**480.3**	**612.1**	**566.1**	951.8	1663.0	1212.5	↓	1703.7	2918.9	2311.6	↓	1418.4	1910.8	1589.6	↓
hiva	2393.4	3511.1	**2608.4**	9910.8	15 893.2	13 195.4	↓	11 212.7	18 778.3	14 081.5	↓	1842.8	3231.6	3019.7	↓
arrhythmia	18.2	25.4	**19.0**	38.7	51.3	47.1	↓	25.1	30.9	27.9	↓	17.1	24.3	19.1	=
mf	312.4	418.1	362.1	432.2	380.3	459.7	↓	266.4	414.7	326.2	=	277.0	505.4	364.9	↓
tm1	27.1	30.4	28.0	21.7	20.0	22.3	↑	23.32	20.9	22.1	↑	17.87	23.9	20.8	↑
tm2	**28.3**	**31.2**	**29.0**	32.1	36.4	34.2	↓	32.5	38.7	35.7	↓	29.4	34.4	32.6	↓
tm3	**12.7**	**16.6**	**14.5**	40.1	69.1	60.0	↓	91.4	131.1	124.9	↓	–	–	–	–

In our experiments, method *A* corresponds to the proposed OS-NRRSAR-SA and method B corresponds to one of the grafting, information-investing, and fast-OSFS algorithms. Moreover, in all the tests, the statistical significance level (α) is set to be 0.05.

Table 5 reports the compactness of the selected subsets using the four algorithms. We can conclude that the proposed algorithm selects fewer features than the other three algorithms. For most datasets, except the *madelon* and the last five datasets, grafting and information-investing found subsets which are considerably large. This can be attributed to the nesting effect of the two algorithms. For *madelon*, *arrhythmia*, *mf*, *tm1*, *tm2*, and *tm3* datasets, the nesting effect is not observable, because of the small size of the feature space. For large scale datasets, the fast-OSFS algorithm found smaller subsets in comparison to the grafting and information-investing algorithms. Although the sizes of the subsets found by fast-OSFS algorithm for the *cia0* and *arrhythmia* datasets are comparable with those found by OS-NRRSAR-SA, it lost the comparison for most of the tests. Moreover, the fast-OSFS algorithm failed to select a feature subset for two datasets, *arcene* and *tm3*. This may be due to the limited number of training instances (this algorithm uses conditional independence test, which needs sufficiently large number of training instances).

The run-time results are reported in Table 6. We see that the OS-NRRSAR-SA is superior for eight datasets, *dorothea*, *arcene*, *dexter*, *madelon*, *sido0*, *sylva*, *tm2*, and *tm3*. Comparing OS-NRRSAR-SA with fast-OSFS, show that the fast-OSFS algorithm is superior only in two cases *cina0* and *tm1*. But, this may be related to the fact that during the feature selection process, the fast-OSFS keeps the subset size very small in comparison with the OS-NRRSAR-SA for these particular datasets. Although the minimum rum-times in *nova*, *hiva*, *arrhythmia* and *mf* are smaller using fast-OSFS, the hypothetical t-tests do not confirm these results. The tests show that the mean run-times of OS-NRRSAR-SA is significantly smaller for *hiva* and *mf*. Moreover, there is not significant evidence that the mean run-times of fast-OSFS is different from OS-NRRSAR-SA for *nova* and *arrhythmia*. An other important result is that the grafting and information-investing algorithms are not time-efficient for data-sets with large feature space.

The classification results, presented in Tables 7–10, show that the OS-NRRSAR-SA algorithm performs very well and shows increase in classification accuracies for most of the tests.

Compared with grafting in terms of J48 classifier (Table 7), the OS-NRRSAR-SA is superior in all tests, except the *tms* datasets. The grafting won the test for *tm1* and *tm2*, but the hypothetical t-test shows that there is no significant difference

Table 7
J48 classification results at the end of the feature selection process for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**89.10**	**94.02**	**92.16**	36.65	51.39	43.30	↑	34.65	51.69	43.36	↑	81.90	91.19	87.34	↑
arcene	**84.70**	**90.62**	**87.77**	56.00	72.61	68.62	↑	53.56	59.19	55.16	↑	–	–	–	–
dexter	**88.20**	**95.41**	**91.10**	79.27	85.06	82.50	↑	52.13	64.90	57.62	↑	78.72	88.54	85.42	↑
madelon	**65.25**	**83.27**	74.39	28.53	40.66	32.18	↑	34.04	44.83	39.12	↑	64.89	83.19	74.55	=
sido0	**96.25**	**100**	**98.20**	22.00	56.93	36.28	↑	19.73	55.27	29.13	↑	86.28	91.46	89.83	↑
cina0	57.19	**84.46**	**66.72**	42.18	68.73	60.99	↑	39.52	72.54	46.27	↑	62.63	80.14	65.36	=
nova	**72.50**	**86.05**	**80.46**	48.35	71.13	58.92	↑	53.28	66.65	60.58	↑	55.93	85.32	72.33	↑
sylva	**86.47**	**90.24**	**88.22**	17.23	34.72	25.48	↑	25.84	52.14	38.81	↑	34.71	50.03	41.44	↑
hiva	**51.42**	**62.29**	**55.78**	13.80	24.12	21.80	↑	17.77	31.12	22.19	↑	12.83	44.48	19.11	↑
arrhythmia	**90.63**	**97.09**	**95.36**	62.18	85.72	80.12	↑	76.62	87.00	83.28	↑	83.55	93.12	89.11	↑
mf	82.36	88.93	84.72	51.88	72.18	67.63	↑	84.33	95.12	89.32	↓	81.36	90.33	88.63	=
tm1	80.19	84.02	82.89	96.36	100	97.98	↓	80.19	85.00	82.77	=	82.12	88.34	84.19	↓
tm2	90.73	**98.53**	**93.74**	90.73	97.99	93.37	=	87.63	94.48	91.34	↑	82.46	93.48	85.73	↑
tm3	39.82	65.26	51.72	55.81	81.77	71.29	↓	41.82	66.81	50.96	=	–	–	–	–

Table 8
JRip classification results at the end of the feature selection process for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**93.45**	**98.86**	**96.63**	21.86	42.98	30.86	↑	30.65	45.00	37.02	↑	72.49	91.19	82.66	↑
arcene	**82.54**	**90.88**	**87.91**	56.00	77.33	71.32	↑	59.69	69.04	61.48	↑	–	–	–	–
dexter	**92.62**	**98.00**	**95.26**	59.33	77.84	71.35	↑	46.33	60.15	50.74	↑	72.65	85.66	80.90	↑
madelon	**65.20**	**84.00**	**74.81**	32.53	40.66	37.10	↑	37.27	50.17	44.92	↑	63.90	83.19	74.11	=
sido0	**96.25**	**100**	**98.33**	21.25	37.64	31.93	↑	20.66	50.28	32.90	↑	85.72	90.47	89.02	↑
cina0	48.83	75.01	63.50	34.29	61.75	49.62	↑	34.07	52.11	39.98	↑	60.66	79.55	64.58	↓
nova	**77.18**	**91.88**	**86.11**	36.44	71.47	53.99	↑	54.66	66.43	61.09	↑	55.93	85.32	72.33	↑
sylva	**77.53**	**84.03**	**79.80**	19.71	35.60	26.84	↑	27.05	50.10	34.55	↑	34.71	46.12	40.94	↑
hiva	**52.77**	**61.86**	**55.70**	15.77	23.73	20.03	↑	17.82	35.82	23.19	↑	16.33	30.41	17.03	↑
arrhythmia	**91.00**	**94.54**	**93.77**	63.55	80.03	76.44	↑	79.60	87.67	82.24	↑	84.45	93.00	85.04	↑
mf	**83.71**	88.12	85.84	49.01	64.82	63.18	↑	80.91	93.00	90.32	↓	77.80	88.10	81.82	↑
tm1	83.43	87.12	85.91	95.07	100	97.74	↓	78.30	86.82	83.11	↑	82.88	86.17	85.21	=
tm2	90.19	96.00	93.05	91.27	98.03	93.43	=	84.90	90.32	88.63	↑	83.00	93.48	86.44	↑
tm3	33.54	60.17	47.11	56.00	85.03	76.11	↓	41.85	69.49	59.61	↓	–	–	–	–

Table 9
SVM classification results at the end of the feature selection process for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**95.42**	**96.77**	**96.00**	22.16	52.44	35.40	↑	36.02	65.15	47.97	↑	84.82	89.44	87.24	↑
arcene	**91.04**	**95.18**	**91.86**	31.88	48.96	42.46	↑	44.44	71.37	59.35	↑	–	–	–	–
dexter	**90.65**	**96.50**	**94.14**	63.01	81.71	71.39	↑	60.32	72.67	67.54	↑	81.44	93.44	88.28	↑
madelon	**81.95**	**89.61**	**85.80**	47.37	66.51	58.06	↑	63.29	72.57	66.83	↑	41.90	59.02	49.63	↑
sido0	**92.14**	**95.16**	**94.03**	51.73	72.83	66.12	↑	53.90	57.63	55.18	↑	66.73	80.83	74.44	↑
cina0	**63.88**	**88.19**	**79.03**	61.82	84.49	73.18	↑	24.90	37.19	30.91	↑	52.52	59.12	57.46	↑
nova	**89.73**	**94.03**	**92.22**	29.66	44.00	37.81	↑	12.25	26.88	20.01	↑	82.90	90.62	87.79	↑
sylva	**92.82**	**99.01**	**94.88**	40.66	61.11	53.92	↑	65.00	69.95	65.70	↑	19.45	43.88	31.95	↑
hiva	**45.43**	**60.36**	**52.00**	22.01	30.22	25.91	↑	9.99	15.28	13.17	↑	21.54	53.48	33.84	↑
arrhythmia	*90.98*	91.05	90.99	87.72	98.34	93.12	↓	77.12	86.82	82.03	↑	89.33	92.95	90.63	=
mf	62.88	73.19	69.03	56.93	69.12	63.06	↑	90.44	97.78	92.91	↓	81.90	89.19	86.92	↓
tm1	89.38	91.23	90.83	94.44	96.00	95.32	↓	56.03	66.83	60.23	↑	44.20	62.19	56.66	↑
tm2	**94.55**	**98.54**	**96.44**	77.65	95.99	86.23	↑	84.11	94.22	89.31	↑	54.12	60.33	58.42	↑
tm3	65.99	68.03	67.32	80.19	90.66	84.94	↓	75.99	83.15	80.00	↓	–	–	–	–

between the results of the two algorithms for *tm2*. Moreover, OS-NRRSAR-SA shows an increase of up to 60 percent for *sido0* and *sylva*, up to 45 percent for *dorothea* and *madelon*, and up to 35 percent for *hiva*. Comparing the two algorithms in terms of JRip (Table 8), we see the similar results to those discussed for J48. Using SVM (Table 9), the grafting shows slightly better results for the *arrhythmia*, *tm1*, and *tm3* datasets, but we can see that our proposed algorithm is superior in the other eleven datasets and it shows an increase of up to 60 percent for *dorothea* and up to 50 percent for *arcene* and *nova* datasets. In terms of Naive Bayes classifier (Table 10), OS-NRRSAR-SA won the tests for ten datasets, while it lost for three cases.

Table 10
Naive Bayes classification results at the end of the feature selection process for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA			Grafting				Information-investing				fast-OSFS			
	min	max	avg	min	max	avg	t	min	max	avg	t	min	max	avg	t
dorothea	**84.32**	**84.90**	**84.65**	19.44	34.45	28.04	↑	26.39	44.30	34.84	↑	71.12	82.19	78.74	↑
arcene	**76.11**	**88.56**	**81.65**	42.12	54.00	47.12	↑	16.10	18.13	16.87	↑	–	–	–	–
dexter	**86.10**	**91.92**	**88.30**	50.68	59.55	53.93	↑	65.12	74.89	67.32	↑	60.11	82.05	76.10	↑
madelon	**93.50**	**96.00**	**94.48**	32.19	49.05	45.12	↑	69.55	84.50	79.78	↑	76.44	77.00	76.80	↑
sido0	**94.90**	**100**	**98.63**	43.10	55.82	49.99	↑	56.11	65.01	60.50	↑	80.80	91.44	86.98	↑
cina0	56.11	72.50	67.12	82.90	88.23	84.77	↓	15.22	34.10	26.48	↑	68.12	79.22	73.92	↓
nova	**96.28**	**100**	**98.90**	45.67	64.55	56.87	↑	34.33	56.99	49.87	↑	66.59	80.00	75.23	↑
sylva	**98.11**	**100**	**98.82**	72.12	88.46	81.14	↑	32.73	36.00	34.11	↑	60.19	68.50	65.19	↑
hiva	**30.66**	**41.25**	**36.21**	11.55	18.28	14.85	↑	16.44	17.00	16.65	↑	17.43	23.22	19.54	↑
arrhythmia	**80.12**	**85.00**	**82.19**	78.10	85.12	82.37	=	77.12	86.82	82.03	↑	81.56	83.50	81.77	=
mf	**70.41**	**77.45**	**73.54**	28.53	41.19	34.88	↑	52.03	66.80	57.98	↑	25.67	54.11	40.05	↑
tm1	**93.19**	**96.11**	**94.00**	67.33	85.48	77.37	↑	44.28	57.00	52.50	↑	79.23	90.39	85.03	↑
tm2	94.55	98.76	96.38	97.88	100	99.03	↓	70.33	82.13	76.66	↑	90.49	95.29	92.88	↑
tm3	46.09	49.03	46.03	60.44	80.45	73.19	↓	55.20	72.02	64.92	↓	–	–	–	–

Compared with information-investing in terms of J48, our proposed algorithm is superior in eleven tests, while the results of the two algorithms are comparable for the three remaining tests (*mf*, *tm2*, and *tm3*). Moreover, we can see that OS-NRRSAR-SA shows an increase of up to 70 percent for *sido0*, and up to 50 percent for *dorothea* and *sylva*. Using JRip, OS-NRRSAR-SA lost the tests for two datasets, *mf* and *tm3*. However, we can see that this algorithm shows an increase of up to 60 percent for two datasets, *dorothea* and *sido*. Using SVM, we can see the similar results to those discussed for JRip. Comparing the two algorithms in terms of Naive Bayes, we can see that OS-NRRSAR-SA lost the tests only for *tm3* dataset.

Comparing with fast-OSFS in terms J48 classifier, the OS-NRRSAR-SA algorithm is superior in eight tests (the *arcene* and *tm3* are not considered). Our algorithm lost the tests only for *tm1*, and there is no significant difference between the results of the two algorithms for *madelon*, *cina0*, and *mf*. Moreover, OS-NRRSAR-SA exhibits an increase of up to 45 percent for *sylva* and *hiva* datasets. Comparing the two algorithms in terms of JRip, the OS-NRRSAR-SA algorithm is superior in nine tests and it lost only for *cina0*. Our proposed algorithm exhibited better results using SVM and Naive Bayes and won the tests for ten tests in each classifier.

Fig. 9 represents the average classification accuracy of selected subsets during features streaming using J48 classifier for the 30 random streaming orders. A general conclusion from this figure is that the proposed OS-NRRSAR-SA algorithm is more accurate at each time instance and therefore, allows reliable classification and learning tasks at that time instance during the streaming phase. Moreover, the grafting and information-investing algorithms show lowest changes in classification accuracies as more features are seen. This can be attributed to the lower dynamism of the two algorithms, which is due to the nesting effect. In Table 11, we report the average dynamism of the four algorithms for the 30 streaming orders. The dynamism of the selected subset is calculated as the number of the last streaming feature that changed the selected subset to the total number of features. As it can be seen from this table, the grafting and information-investing algorithms ignore the majority of the feature space for most of the large scale datasets. The learners become more and more complex as new features stream in and therefore, the features which appear late are not able to increase the model accuracy.

5.1.2. Comparison of OS-NRRSAR-SA with DIA-RED

In this subsection, the effectiveness and efficiency of the OS-NRRSAR-SA is compared with the DIA-RED algorithm [50]. DIA-RED maintains a rough sets-based entropy value of the current selected subsets and updates this value whenever new conditional features are added to the dataset. DIA-RED is implemented using the combination entropy [42] and the size of incremental attribute set (SIA) is set to be 1.

Table 12 reports the compactness, run-time, and accuracy comparisons of the two algorithms for four lower dimensional datasets, *madelon*, *arrhythmia*, *tm1*, and *tm2*. The results are for a single streaming order. Moreover, DIA-RED is not time tractable for the remaining ten datasets in Table 4, and therefore the results are not reported. As it can be seen from Table 12, the proposed OS-NRRSAR-SA is superior in all comparison terms. DIA-RED does not implement an effective redundant attribute elimination mechanism, and therefore the selected subsets are large during features streaming. This causes ineffective partitioning steps in calculating the rough sets approximations, and therefore, this algorithm is not time efficient. The low accuracy of DIA-RED may be due to the fact that this algorithm uses only the information contained within the lower approximation of a set ignoring the information contained in the boundary region.

5.1.3. Robustness against noise

Here, we investigate the effect(s) of noise on selected subsets of OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS. In this regard, several levels of noises are added to the datasets and then the four feature selection algorithms are applied again on these datasets. In order to add a noise with probability K to a decision system $D = (U, F, d)$, we changed $f_i(x)$, $i = 1, 2, \ldots, |F|$, $x = 1, 2, \ldots, |U|$, with probability K, as follows

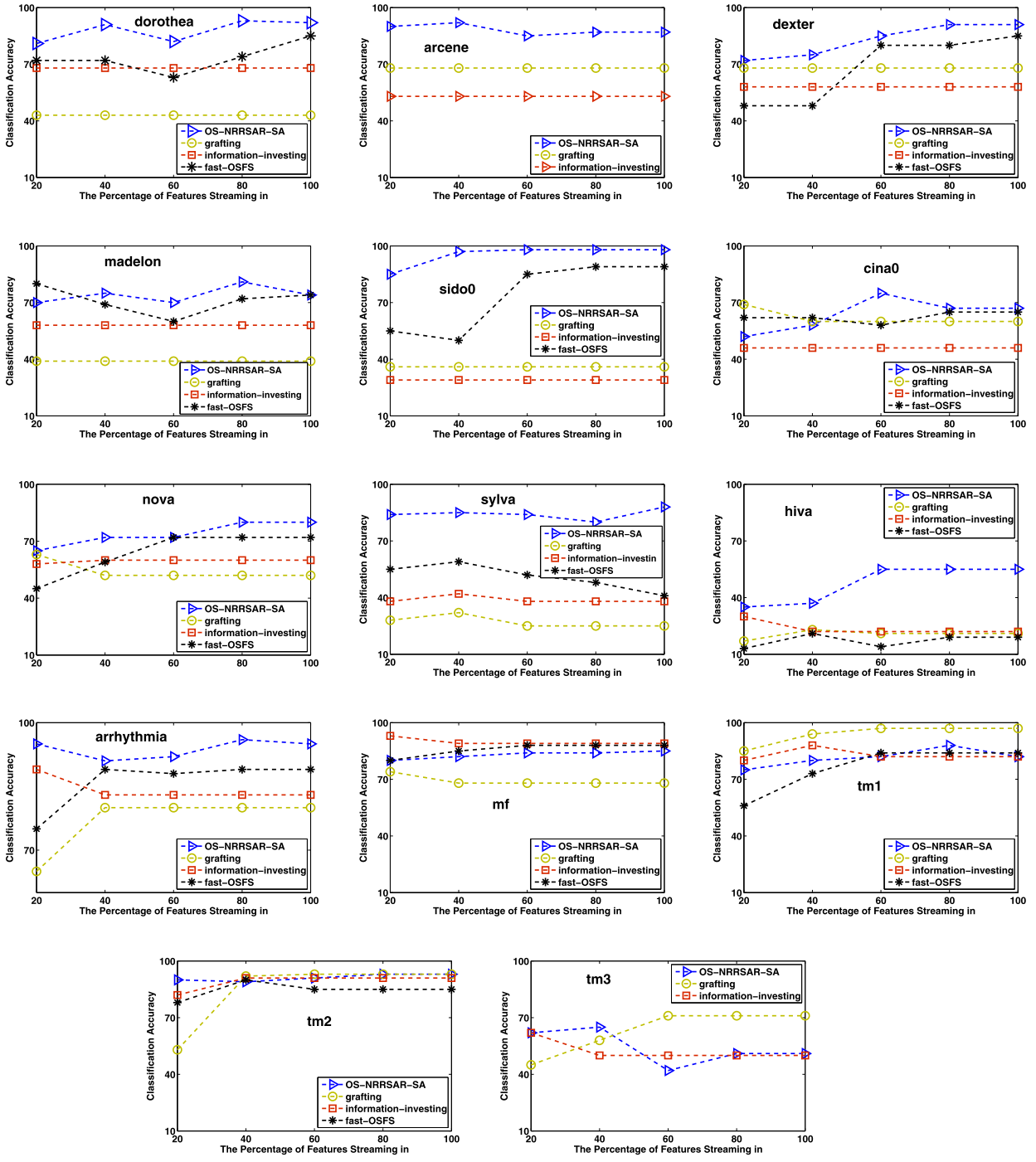

Fig. 9. Classification results of selected subsets during features streaming.

- If f_i is categorical or integer-valued feature, with value set (domain) V_{f_i}, then we replaced $f_i(x)$ with a randomly selected value from $V_{f_i} - \{f_i(x)\}$.
- If f_i is a real-valued feature, then we first normalized $f_i(x)$ using the following formula:

$$f_i^{(n)}(x) = \frac{f_i(x) - \min(f_i)}{\max(f_i) - \min(f_i)} \tag{29}$$

Table 11
Comparison of dynamism for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS.

Dataset	OS-NRRSAR-SA	Grafting	Information-investing	fast-OSFS
dorothea	78.3%	9.4%	14.8%	60.6%
arcene	84.5%	29.4%	15.1%	–
dexter	78.0%	11.8%	8.9%	49.2%
madelon	88.9%	12.7%	9.3%	70.5%
sido0	97.2%	18.6%	16.2%	74.2%
cina0	69.1%	27.0%	20.3%	60.1%
nova	97.8%	16.0%	22.0%	69.7%
sylva	90.3%	88.4%	75.2%	81.9%
hiva	91.2%	46.2%	32.4%	73.0%
arrhythmia	87.3%	21.6%	31.9%	60.7%
mf	91.3%	30.7%	22.9%	60.0%
tm1	67.0%	62.1%	46.2%	77.1%
tm2	83.4%	67.5%	49.4%	43.9%
tm3	66.9%	29.3%	35.9%	-

Table 12
Comparison of OS-NRRSAR-SA and DIA-RED.

Dataset	Run-time		Selected subset size		Accuracy (J48)	
	OS-NRRSAR-SA	DIA-RED	OS-NRRSAR-SA	DIA-RED	OS-NRRSAR-SA	DIA-RED
madelon	63.19	11 283	4	61	79.28	62.03
arrhythmia	23.72	6392	9	46	95.89	32.22
tm1	29.11	7783	6	29	82.01	29.45
tm2	30.47	8035	6	37	94.37	41.38

Then we added a noise to $f_i^{(n)}(x)$ as

$$g_i(x) = f_i^{(n)}(x) + \mathcal{N}(0, 0.3) \tag{30}$$

where $\mathcal{N}(0, 0.3)$ is a gaussian noise with mean 0 and variance 0.3. Finally we re-scaled $g_i(x)$ using the following formula:

$$f_i(x) = g_i(x)\,(\max(f_i) - \min(f_i)) + \min(f_i) \tag{31}$$

For each algorithm A and decision system D, we define two functions Δ_A^D and Θ_A^D as

$$\Delta_A^D = \left| \frac{R_A^{(D^{(n)})} - R_A^{(D)}}{R_A^{(D^{(n)})}} \right| \tag{32}$$

$$\Theta_A^D = \left| \frac{R_A^{(D)} \cap R_A^{(D^{(n)})}}{R_A^{(D)}} \right| \tag{33}$$

Where,
A: an algorithm from {OS-NRRSAR-SA, grafting, information-investing, fast-OSFS},
$D^{(n)}$: the decision system D which is disrupted using noise,
$R_A^{(D)}$: the set of selected features using algorithm A for decision system D.

$\Theta_A^D \in [0, 1]$ is a measure of how much algorithm A is able to select those features for $D^{(n)}$, which are selected for D. Moreover, $\Delta_A^D \in [0, 1]$ is a measure of how much algorithm A selects features for $D^{(n)}$, which are not selected for D.

For each dataset, we generated 5 different streaming orders randomly. Then for each order we applied the comparing feature selection algorithms twice; once on the original dataset and once again on the corresponding disrupted dataset. We have used two levels of noises, $K = 0.05$, and $K = 0.1$. Fig. 10 reports the Θ and Δ values recorded for OS-NRRSAR-SA, grafting, information-investing, and fast-OSFS. The results for each algorithm and dataset are averaged over the five streaming orders. As it can be seen from this figure, noises have least effect on results of the proposed OS-NRRSAR-SA algorithm, while the other three algorithms are very noise-sensitive and their results are destroyed, even using small levels of noises.

6. Conclusions

Feature selection, as a pre-processing step, is to select a small subset of most important and discriminative input features. This paper considered the OSFS problem from the rough sets (RS) perspective and the main motivation for this consideration

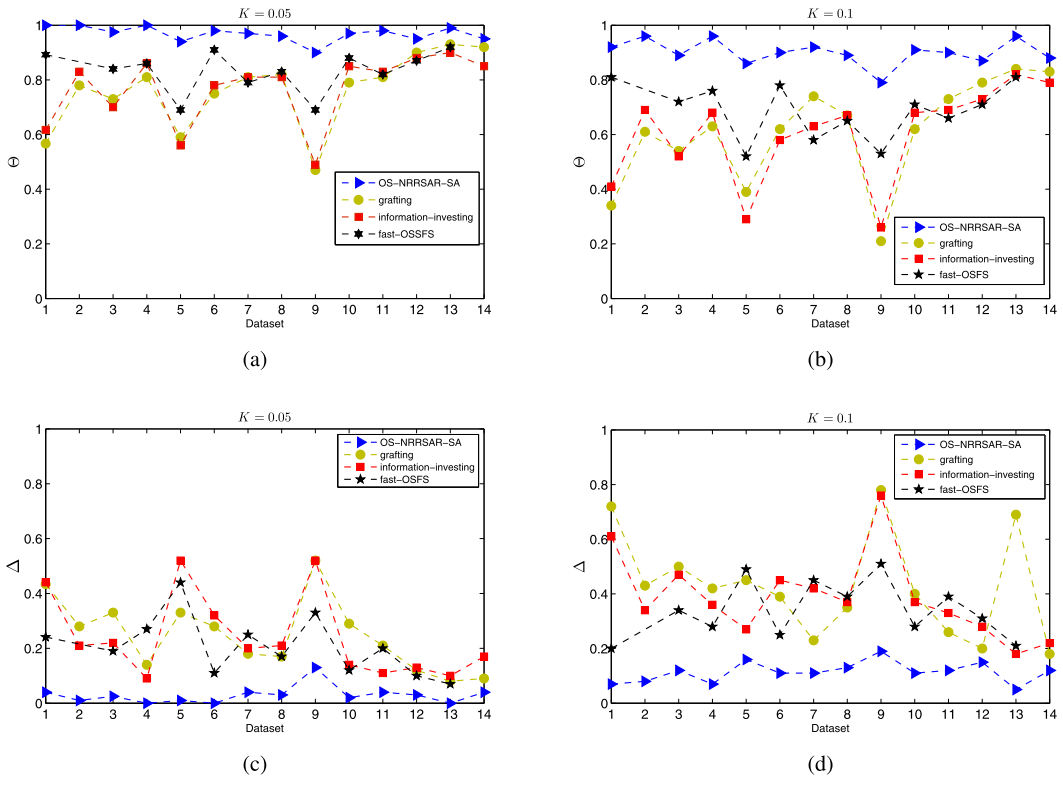

Fig. 10. Recorded Θ and Δ values for OS-NRRSAR-SA, Grafting, Information-Investing, and fast-OSFS. a) Θ values for $K = 0.05$, b) Θ values for $K = 0.1$, c) Δ values for $K = 0.05$, d) Δ values for $K = 0.1$. (The labels of the x-axis from 1 to 14 denote the datasets: 1: dorothea; 2: arcene; 3: dexter; 4: madelon; 5: sido0; 6: cina0; 7: nova; 8: sylva; 9: hiva; 10: arrhythmia; 11: mf; 12: tm1; 13: tm2; 14: tm3).

was that RS-based data mining do not require any domain knowledge other than the given dataset. In the process of developing a RS-based OSFS algorithm, a new OSFS algorithms, called OS-NRRSAR-SA, is proposed. OS-NRRSAR-SA adopts the classical feature significance concept to eliminate features which have no influence in deciding output feature. The main advantage of the proposed algorithm is that it does not require any domain knowledge about feature space and therefore, it is an excellent choice for true OSFS scenarios. To show the efficiency and accuracy of the proposed algorithm, it is compared with grafting, information-investing, fast-OSFS, and DIA-RED algorithms. Several high dimensional datasets are used for comparisons, and their features considered one by one to simulate the true OSF scenarios. The compactness, run-time, classification accuracy at the end of the streaming, and classification accuracy of selected subsets during features streaming, were the comparison terms. The experiments demonstrate that the proposed algorithm achieves better results than existing OSFS algorithms, for all evaluation terms. Additional comparisons have shown that noises have least effect on results of the proposed OS-NRRSAR-SA algorithm, while the other three algorithms are very noise-sensitive and even small levels of noises have significant effects on their results.

References

[1] R. Battiti, Using mutual information for selecting features in supervised neural net learning, IEEE Trans. Neural Netw. 5 (4) (1994) 537–550.
[2] C. Blake, C.J. Merz, UCI repository of machine learning databases, 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998 [Online; accessed 06-March-2015].
[3] G. Brown, A. Pocock, M.-J. Zhao, M. Luján, Conditional likelihood maximisation: a unifying framework for information theoretic feature selection, J. Mach. Learn. Res. 13 (1) (2012) 27–66.
[4] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27:1–27:27.
[5] G. Chen, J. Chen, A novel wrapper method for feature selection and its applications, Neurocomputing 159 (2015) 219–226.
[6] Y. Chen, A. Abraham, B. Yang, Feature selection and classification using flexible neural tree, Neurocomputing 70 (1–3) (2006) 305–313.
[7] Clopinet, Feature selection challenge, NIPS 2003, http://clopinet.com/isabelle/Projects/NIPS2003/, 2003 [Online; accessed 06-March-2015].
[8] Clopinet, Performance prediction challenge, WCCI 2006, http://clopinet.com/isabelle/Projects/modelselect/, 2006 [Online; accessed 06-March-2015].
[9] Clopinet, Causation and prediction challenge, WCCI 2008, http://www.causality.inf.ethz.ch, 2008 [Online; accessed 06-March-2015].
[10] W.W. Cohen, Fast effective rule induction, in: Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 115–123.
[11] M. Dash, H. Liu, Consistency-based search in feature selection, Artif. Intell. 151 (1–2) (2003) 155–176.
[12] P. Domingos, M. Pazzani, On the optimality of the simple bayesian classifier under zero–one loss, Mach. Learn. 29 (2–3) (1997) 103–130.
[13] D. Dubois, H. Prade, Putting rough sets and fuzzy sets together, in: R. Słowiński (Ed.), Intelligent Decision Support, in: Theory and Decision Library, vol. 11, Springer, Netherlands, 1992, pp. 203–232.

[14] S. Eskandari, M.M. Javidi, A noise resistant dependency measure for rough sets based feature selection, 2015, unpublished.
[15] K. Glocer, D. Eads, J. Theiler, Online feature selection for pixel classification, in: Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 2005.
[16] I. Guyon, A. Elliseff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
[17] A.E. Hassanien, Rough set approach for attribute reduction and rule generation: a case of patients with suspected breast cancer, J. Am. Soc. Inf. Sci. Technol. 55 (11) (2004) 954–962.
[18] A.R. Hedar, J. Wang, M. Fukushima, Tabu search for attribute reduction in rough set theory, Soft Comput. 12 (9) (2008) 909–918.
[19] S.C.H. Hoi, J. Wang, P. Zhao, R. Jin, Online feature selection for mining big data, in: Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, BigMine '12, ACM, New York, NY, USA, 2012, pp. 93–100, URL http://doi.acm.org/10.1145/2351316.2351329.
[20] F. Hu, G. Wang, H. Huang, Y. Wu, Incremental attribute reduction based on elementary sets, in: D. Ślęzak, G. Wang, M. Szczuka, I. Düntsch, Y. Yao (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, in: Lecture Notes in Computer Science, vol. 3641, Springer, Berlin, Heidelberg, 2005, pp. 185–193.
[21] Z. Hu, Y. Bao, T. Xiong, R. Chiong, Hybrid filter–wrapper feature selection for short-term load forecasting, Eng. Appl. Artif. Intell. 40 (2015) 17–27.
[22] R. Jensen, Qiang Shen, Semantics-preserving dimensionality reduction: rough and fuzzy–rough based approaches, IEEE Trans. Knowl. Data Eng. 16 (16) (2004) 1457–1471.
[23] R. Jensen, Q. Shen, A rough set-aided system for sorting www bookmarks, in: Proceedings of the First Asia–Pacific Conference on Web Intelligence: Research and Development, WI '01, London, UK, 2001.
[24] R. Jensen, Q. Shen, Fuzzy–rough data reduction with ant colony optimization, Fuzzy Sets Syst. 149 (1) (2005) 5–20.
[25] R. Jensen, Q. Shen, Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches, IEEE Press Series on Computational Intelligence, Wiley, 2008.
[26] R. Jensen, A. Tuson, Q. Shen, Finding rough and fuzzy–rough set reducts with SAT, Inf. Sci. 255 (2014) 100–120.
[27] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artif. Intell. 97 (1–2) (1997) 273–324.
[28] G. Lang, Q. Li, M. Cai, T. Yang, Q. Xiao, Incremental approaches to knowledge reduction based on characteristic matrices, Int. J. Mach. Learn. Cybern. (2014) 1–20.
[29] H.R. Li, W.X. Zhang, Applying indiscernibility attribute sets to knowledge reduction, in: Advances in Artificial Intelligence, AI 2005, vol. 3809, 2005, pp. 816–821.
[30] K. Li, Y.S. Liu, Rough set based attribute reduction approach in data mining, in: 2002 International Conference on Machine Learning and Cybernetics. Proceedings. vol. 1, 2002, pp. 60–63.
[31] T. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowl.-Based Syst. 20 (5) (2007) 485–494.
[32] J. Liang, F. Wang, C. Dang, Y. Qian, A group incremental approach to feature selection applying rough set technique, IEEE Trans. Knowl. Data Eng. 26 (2) (Feb 2014) 294–308.
[33] R. Liu, Y. Shi, Spatial distance join based feature selection, Eng. Appl. Artif. Intell. 26 (10) (2013) 2597–2607.
[34] S. Maldonado, R. Weber, A wrapper method for feature selection using support vector machines, Inf. Sci. 179 (13) (2009) 2208–2217.
[35] M. Modrzejewski, Feature selection using rough sets theory, in: Proceedings of the European Conference on Machine Learning, ECML '93, London, UK, UK, 1993, pp. 213–226.
[36] N. Parthalain, Q. Shen, R. Jensen, A distance measure approach to exploring the rough set boundary region for attribute reduction, IEEE Trans. Knowl. Data Eng. 22 (3) (March 2010) 305–317.
[37] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (5) (1982) 341–356.
[38] H. Peng, F. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1238.
[39] S. Perkins, K. Lacker, J. Theiler, Grafting: fast, incremental feature selection by gradient descent in function space, J. Mach. Learn. Res. 3 (2003) 1333–1356.
[40] S. Perkins, J. Theiler, Online feature selection using grafting, in: International Conference on Machine Learning, ACM Press, 2003, pp. 592–599.
[41] P. Pudil, J. Novovičová, J. Kittler, Floating search methods in feature selection, Pattern Recognit. Lett. 15 (11) (1994) 1119–1125.
[42] Y. Qian, J. Liang, Combination entropy and combination granulation in rough set theory, Int. J. Uncertain. 16 (2) (2008) 179–193.
[43] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
[44] A. Skowron, J. Stepaniuk, Tolerance approximation spaces, Fundam. Inform. 27 (2–3) (1996) 245–253.
[45] R.W. Swiniarski, A. Skowron, Rough set methods in feature selection and recognition, Pattern Recognit. Lett. 24 (6) (2003) 833–849.
[46] S. Tabakhi, P. Moradi, F. Akhlaghian, An unsupervised feature selection algorithm based on ant colony optimization, Eng. Appl. Artif. Intell. 32 (2014) 112–123.
[47] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Academic Press, 2009.
[48] L. Ungar, J. Zhou, D. Foster, B. Stine, Streaming feature selection using IIC, in: Proceedings of the 10th International Conference on Artificial Intelligence and Statistics, 2005.
[49] B. Walczak, D. Massart, Rough sets theory, Chemom. Intell. Lab. Syst. 47 (1) (1999) 1–16.
[50] F. Wang, J. Liang, Y. Qian, Attribute reduction: a dimension incremental strategy, Knowl.-Based Syst. 39 (2013) 95–108.
[51] J. Wang, P. Zhao, S. Hoi, R. Jin, Online feature selection and its applications, IEEE Trans. Knowl. Data Eng. 26 (3) (2014) 698–710.
[52] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second edition, Morgan Kaufmann Series in Data Management Systems, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.
[53] X. Wu, K. Yu, W. Ding, H. Wang, X. Zhu, Online feature selection with streaming features, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 1178–1192.
[54] Z. Zeng, H. Zhang, R. Zhang, C. Yin, A novel feature selection method considering feature interaction, Pattern Recogn. 48 (8) (2015) 2656–2666.
[55] P. Zhao, B. Yu, On model selection consistency of lasso, J. Mach. Learn. Res. 7 (2006) 2541–2563.
[56] N. Zhong, J. Dong, S. Ohsuga, Using rough sets with heuristics for feature selection, J. Intell. Inf. Syst. 16 (3) (2001) 199–214.
[57] J. Zhou, D. Foster, R. Stine, L. Ungar, Streaming feature selection using alpha-investing, in: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, Chicago, IL, USA, 2005.
[58] W. Ziarko, Variable precision rough set model, J. Comput. Syst. Sci. 46 (1) (1993) 39–59.