Int. J. Mach. Learn. & Cyber. (2018) 9:667–676
https://doi.org/10.1007/s13042-016-0595-y

ORIGINAL ARTICLE

Streamwise feature selection: a rough set method

Mohammad Masoud Javidi[1] · **Sadegh Eskandari**[1]

Received: 28 July 2015 / Accepted: 12 September 2016 / Published online: 19 September 2016

Abstract Traditional feature selection methods assume that the entire input feature set is available from the beginning. However, streaming features (SF) is an integral part of many real-world applications. In this scenario, the number of training examples is fixed while the number of features grows with time as new features stream in. A critical challenge for streamwise feature selection (SFS) is the unavailability of the entire feature set before learning starts. Several efforts have been made to address the SFS problem, however they all need some prior knowledge about the entire feature set. In this paper, the SFS 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. The proposed method uses the significance analysis concepts in RS theory to control the unknown feature space in SFS problems. This algorithm is evaluated extensively on several high-dimensional datasets in terms of compactness, classification accuracy, and running time. Experimental results demonstrate that the algorithm achieves better results than existing SFS algorithms.

Keywords Feature selection · Streamwise feature selection · Rough sets theory · Significance

✉ Sadegh Eskandari
eskandari@math.uk.ac.ir

Mohammad Masoud Javidi
javidi@uk.ac.ir

[1] Shahid Bahonar University of Kerman, Kerman, Iran

1 Introduction

Given a dataset, the task of feature selection is to select the smallest subset of the most important and discriminative input features. The aim of feature selection is to overcome the curse of dimensionality, which is a severe difficulty that can arise in datasets of high dimensions [1–3].

All the traditional feature selection methods, assume that the entire input feature set is available from the beginning. However, streaming features (SF) is an integral part of many real-world applications. In SF, the number of feature vectors (instances) is fixed while feature set (attributes) grows with time. For example, in bio-informatic and clinical machine learning problems, acquiring the entire set of features for every training instance is expensive due to the high cost lab experiments [4]. 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 [5]. In all these scenarios, we need to incrementally update the feature set as new features are available over time. There are also some scenarios where the entire feature space is accessible, but feature streaming offers many advantages. This scenario is common in statistical relational learning [6] and social network analysis [7], where the feature space is very large and exhaustive store and search over the entire feature space is infeasible.

Streamwise feature selection (SFS) is the task of selecting a best feature subset in SF scenarios. Any SFS method must satisfy three critical conditions; Firstly, it should not require any domain knowledge about feature space, because the full feature space is unknown or inaccessible. Secondly, it should allow efficient incremental updates in selected features, specially, when we have a limited amount of computational time available in between

each feature arrival. Finally, it should be as accurate as possible at each time instance to allow reliable classification and learning tasks at that time instance.

Motivated by these challenges, several research efforts have been made to address SFS. Perkins and Theiler proposed an iterative gradient descent algorithm, called grafting [8, 9]. In 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 value for λ requires information about the global feature space. (2) This algorithm suffers from the so-called nesting effect, if a previously chosen feature is later found to be redundant, there is no way for it to be discarded [10].

Ungar et al. [6] proposed a streamwise regression algorithm, called information-investing. In this algorithm, a newly generated feature is added to the model if the entropy reduction is greater than the cost of coding. 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.

Wu et al. [5] proposed a causality-based SFS 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 SF, 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.

Wang et al. [11] 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 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.

In this paper, the SFS 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. Several successful RS-base feature selection algorithms are proposed in the literatures [12–17]. However, all these algorithms consider the batch feature selection problem and are not applicable to SF scenarios. In this paper, a new SFS algorithm, which adopts the classical RS-based feature significance concept to eliminate irrelevant features, is proposed. The efficiency and accuracy of the proposed algorithm is demonstrated using several experimental results.

The remainder of the paper is organized as follows: Sect. 2 summarizes the theoretical background of rough sets along with a look at the rough set-based attribute reduction methods. Section 3 discusses the new SFS algorithm. Section 4 reports experimental results and Sect. 5 concludes the paper.

2 ROUGH SETS

Uncertainty is a natural phenomenon in machine learning, which can be embedded in the entire process of data preprocessing, learning and reasoning [18–20]. Rough sets theory has introduced by Pawlak [21] to express uncertainty by means of boundary region of a set. The main idea of rough set is the use of a known knowledge in knowledge base to approximate the inaccurate and uncertain knowledge [22]. Therefore, the main advantage of this theory is that it requires no human input or domain knowledge other than the given dataset. This section summarizes the theoretical background of rough sets theory along with a look at the rough set-based attribute reduction methods.

2.1 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 $f : U \rightarrow V_f$, for every $\mathrm{f} \in \mathrm{F}$. The set $\mathrm{V_f}$ is called the value set or domain of f. A decision system is a pair $\mathrm{IS} = (\mathrm{U}, \mathrm{F}, \mathrm{d})$, where d is decision feature.

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

$$INDIS(B) = \{(x, y) \in U \times U | \forall f \in B, f(x) = f(y)\}. \quad (1)$$

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 partition of U into B-elementary subsets is denoted by U/B. The time complexity of generating U/B is $(|\mathrm{B}||\mathrm{P}||\mathrm{U}|)$, where $|\mathrm{P}|$ 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 $|\mathrm{U}|$ and therefore the worst-case complexity of generating U/B is $\mathrm{O}(|\mathrm{B}||\mathrm{U}|^2)$ [23]. For most of the real world applications, $|P| \ll |U|$, and therefore, the partitioning process is very time efficient from application view point [23].

2.2 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:

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

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

The $\underline{B}X$ and $\bar{B}X$ approximations define information contained in B. If $\in \underline{B}X$, it certainly belongs to X, but if $x \in \bar{B}X$, it may or may not belong to X.

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

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

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

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

2.3 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 k th degree (denoted $C \Rightarrow_k D$), if

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

where,

$$POS_B(D) = \bigcup_{X \in U/D} \underline{C}X, \tag{8}$$

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.

2.4 Significance

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 [24]

$$\sigma_{(F,d)}(f) = \frac{\gamma(F, d) - \gamma(F - \{f\}, d)}{\gamma(F, d)}. \tag{8}$$

2.5 Reduct

A reduct R is a subset of conditional features that satisfies both of the following conditions:

$$IND_{IS}(R) = IND_{IS}(C), \tag{9}$$

$$\forall R' \subset Rs.t. IND_{IS}(R) \neq IND_{IS}(R'). \tag{10}$$

An optimal reduct is a reduct with minimum cardinality. The intersection of all reducts contains those attributes that cannot be eliminated and is called the core. Finding a minimal reduct is NP-hard [24], because all possible subsets of conditional features must be generated to retrieve such a reduct. Therefore finding a near optimal has generated much of interest. Figure 1 represents the steps of QUICKREDUCT algorithm [25], which searches for a minimal subset without exhaustively generating all possible subsets.

2.6 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) only operates effectively with datasets containing discrete values and therefore it is necessary to perform a discretization step for real-valued attributes [23, 26]. Therefore, several extensions to the original theory have been proposed to deal with real-valued (continuous datasets). Four well known extensions are variable precision rough sets (VPRS) [27], tolerance rough set model (TRSM) [28], fuzzy rough sets (FRS) [29] and neighborhood rough set model (NRSM) [30–32].

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. 1 The QUICREDUCT algorithm [25]

VPRS [27] 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 OS consideration of operating with no domain knowledge.

TRSM [28] 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 OS 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 $\theta(|\mathrm{B}||\mathrm{U}|^2)$, which is equal to worst-case time complexity of the partitioning process in the classical rough sets.

FRS [29] uses fuzzy equivalence classes generated by a fuzzy similarity relation to represent vagueness in data. The term fuzziness refers to the unclear boundary between two linguistic terms and is based on the membership function of fuzzy sets [20, 33]. 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 ($\theta(|B||U|^2)$).

NRSM [30–32] is used to replace the equivalent approximation of traditional rough set model with θ-neighborhood relation, which supports both continuous and discrete datasets. Although successful in dealing with real-valued datasets, NRSM suffers from the low computation efficiency, especially in computing the neighborhood of each record. Moreover, the introduction of a suitable θ value (the distance threshold parameter) is a challenge in this extension, which needs extra knowledge about the whole feature space.

In addition to rough sets extensions, there are also some modifications, which do not change classical rough sets principles. The notable work in this group is the one proposed in [15]. This modification does not redefine the lower and upper approximations in classical rough sets, but introduces a new dependency measure based on the classical rough sets principals to deal with real-valued data. This new dependency measure uses a proximity measure that quantifies the information contained in the boundary region. This measure needs no human input knowledge to deal with available data. Moreover generating equivalence classes in this modification is more efficient than generating tolerance classes and fuzzy equivalence classes in TRSM and FRS, respectively.

3 Streamwise feature selection using rough sets

As stated, in the streaming features (SF), new conditional features flow in one by one over time while the number of objects in dataset remains fixed. In this section, we propose a new algorithm to implement the rough sets theory for feature selection with SF scenarios.

3.1 The proposed algorithm

Because we do not have access to the full feature space in the streaming features context, the batch versions of RS-based feature selection algorithms, such as QUICKREDUCT, are not directly applicable. This problem can be relaxed by allowing a new incoming feature to be included in the selected subset, if it increases the dependency measure. However, this relaxation may be dangerous in SF scenarios, because early incoming features will be selected with more chance and most of the late coming features may not be considered. Here, we first define a generalized definition of significance concept, then, we provide a general rough set-based feature selection algorithm for SF scenarios.

Definition 1 Let $DS = (A, F, d)$ be a decision system and $\pi : 2^{F\cup\{d\}} \times 2^{F\cup\{d\}} \rightarrow [0, 1]$ be a dependency function defined on DS. The generalized significance of $P \in F$, denoted $\sigma^g_{(F,d)}(P)$, is defined as:

$$\sigma^g_{(F,d)}(P) = \frac{\pi(F, d) - \pi(F - P, d)}{\pi(F, d)} \tag{11}$$

π can be any dependency function such as the classical rough set-based dependency function (γ), or a dependency function based on rough sets extensions (such as VPRS, TRSM, and FRS) and modifications (such as the dependency measure defined in [aaa15]).

Definition 2 Let $DS = (A, F, d)$ be a decision system and $P \subseteq F$. P is non-significant for DS, if and only if $\sigma^g_{(F,d)}(P) = 0$.

Definition 3 Let $DS = (A, F, d)$ be a decision system and $P \subseteq F$. DS is π-consistent using P, if and only if $\pi(P, d) = 1$.

Figure 2 represents the proposed streamwise feature selection algorithm, called SFS-RS, which uses generalized significance analysis to control the inclusion of any new incoming feature in SF context. 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 based on the π-consistency of the dataset. If the dataset is not π-consistent ($\pi(R,d) \neq 1$), the algorithm tests the increase of the dependency value, when f is added to current subset (line 5). If the measure is increased, the current subset is updated to include f (line 6), otherwise, f is rejected. On the other hand, if the dataset is π-consistent ($\pi(R,d) = 1$), 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 9–19). If such subset exists, and its size is larger than one, then the subset can be replaced with f (lines 21–23). This makes our dataset smaller, while keeping the π-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 dependency value (lines 23–25). The algorithm alternates the above two phases till the stopping criteria is satisfied. 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, then the algorithm can stop once a predefined accuracy is satisfied or a maximum number of iterations is reached.

SFS-RS (d): Streamwise Feature Selection Using Rough Sets
d: The decision feature

1: $R \leftarrow \{\}$
2: **do**
3: $f \leftarrow GetNewFeature()$
4: **if** $(\pi(R,d) \neq 1)$
5: **if** $(\pi(R \cup \{f\},d) > \pi(R,d))$
6: $R \leftarrow R \cup \{f\}$
7: **end if**
8: **else**
9: /* FIND NON-SIGNIFICANT FEATURES */
10: $B \leftarrow \{\}$
11: $T \leftarrow R$
12: **while** $(|T| \neq 0)$ **do**
13: $g \leftarrow Random(f_i \in T, i = 1, \cdots, |T|)$
14: **if** $(\sigma^{g}{}_{(R,d)}(g) = 0)$
15: $B \leftarrow B \cup \{g\}$
16: **end if**
17: $T \leftarrow T - \{g\}$
18: **end while**
19 /*REMOVE NON-SIGNIFICANT FEATURES */
20: **if** $(|B| > 1)$
21: $R \leftarrow R - B$
22: $R \leftarrow R \cup \{f\}$
23: **end if**
24: **if** $(|B| = 1)$
25: $x \leftarrow \text{argmax}_{a \in \{f\} \cup B}(\pi(a,d))$
26: $R \leftarrow (R \cup \{f\}) - x$
27: **end if**
28: **end if**
29: **until** (stopping criterion is met)

Fig. 2 The proposed streamwise feature selection

3.2 The time complexity of SFS-RS

The time complexity of SFS-RS depends on the number of π tests. Suppose that the time complexity of calculating $\pi(B,d)$ can be attributed by function $\Psi(B,U)$, where U is the set of objects. For example if we use the classical rough set-based dependency function (γ), then $\Psi(B,U) = \Theta(|B||P||U|)$ and if, on the other hand, we use fuzzy rough set-based dependency function, then $\Psi(B,U) = \Theta(|B||U|^2)$. Suppose that at time t a new feature f_t be present to the SFS-RS algorithm and let R_t be the selected feature subset at this time. If the available dataset is not π-consistent using R_t, the first phase of the algorithm will be triggered. This phase includes a single π test and therefore, the worst-case time complexity of this phase is $\Psi(B,U)$. However, if the dataset is π-consistent, the second phase will be triggered, which needs $2|R_t|$ π tests (two π tests for each consistency check) to remove non-significant features. Therefore the worst case time complexity of this phase will be $2|R_t|\Psi(B,U)$.

4 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 SFS algorithm (SFS-RS) with four existing SFS algorithms, grafting [9], information-investing [6], fast-OSFS [5] and DIA-RED [11]. Table 1 summarizes the datasets used in our experiments. The *dorothea*, *arcene*, *dexter*, and *madelon* datasets are from the NIPS 2003 feature selection challenge [34]. *Arrhythmia*, *mf* (multiple features), ionosphere, wine, and credit are selected from the UCI machine learning repository [35], and the *threshold max 1–3* (*tm1–3*) are three synthetic datasets from [9]. All the experiments are carried out on a DELL workstation with Windows 7, 2 GB memory, and 2.4 GHz CPU. The J48 [36] and kernel SVM with RBF kernel function [37] classifiers are used to compare the performance of the proposed SFS algorithm.

4.1 Experiments on different settings of the dependency function

Here we consider the effect of different dependency functions (different settings for π) on the performance of

Table 1 Summary of the Benchmark dimensional datasets

Dataset	#Features	#Instances	Data type
dexter	100000	800	Categorical
arcane	10000	100	Integer
dexter	20000	300	Integer
madelon	500	2000	Categorical
arrhythmia	279	452	Categorical, integer, real
mf	649	2000	Integer, real
tm1	100	1000	Real
tm2	100	1000	Real
tm3	1000	100	Real
ionosphere	34	351	Integer, real
wine	13	178	Integer, real
credit	15	690	Categorical, integer, real

the SFS-RS algorithm. six dependency measures are considered here; (1) The classical rough sets-based dependency function, (2) VPRS-based dependency function, (3) TRSM-based dependency function, (4) FRS-based dependency function, NRSM-based dependency measure, and 6) The dependency measure introduced in [15]. Two different threshold values $\beta = 0.1$ and $\beta = 0.2$ are employed to define the generalized inclusion relation in VPRS-based dependency. Moreover, two different values of tolerance thresholds $\alpha = 0.9$ and $\alpha = 0.95$ are used for TRSM-based function. As experimental results in [32] show that (0.1, 0.3) is an optimal candidate interval for the θ in NRSM, we used the value 0.2 for this parameter. Table 2, summarizes the six dependency measures and their representations used in our reports.

The selected subset sizes, running times, and classification accuracies of the five measures are reported in Tables 3, 4 and 5, respectively. Based on These tables, we can conclude the following results:

- There is only marginal increase in running times of the dependency measure proposed in [15] (ρ in the tables), comparing with the classical rough set-based method. This is because of the fact that the two methods use the same partitioning process to generate elementary classes.
- Although the VPRS-based algorithm is comparable with the classical rough sets-based algorithm, in terms of running times, the results show a strong dependence of the VPRS on the β-value. Although the ideal threshold value can be obtained by repeated experimentation for a given data set, this value will be biased to the used data set. Moreover, finding such a value will impose a large computational time to the overall feature selection process.
- The TRSM, FRS and NRSM based algorithms are very slow and these algorithms are not able to finish for the seven datasets, *dorothea*, *arcene*, *dexter*, *madelon, mf, tm1,* and *tm2*.
- Similar to VPRS, the optimal threshold value in TRSM is data driven and needs a pre-processing step for each data set. This is contrary to the rough sets theory consideration of operating with no domain knowledge.
- Although the classical rough sets-based dependency measure is able to find accurate results for discrete-valued datasets, it lost the tests for most of the continues-valued datasets. Moreover, the dependency proposed in [15] shows increases in classification accuracies for most of the tests, specially, for continues-valued datasets.

Table 2 The five dependency measures and their representations in our experiments

Dependency measure defined by:	Notion
Classical rough sets	γ
Variable precision rough sets	ν
Tolerance rough set model	τ
Fuzzy rough set	ϕ
Neighborhood rough set model	$\mathcal{N}$
[aaa15]	ρ

4.2 Comparison of SFS-RS with other SFS Algorithms

Here, we compare the performance of the proposed SFS algorithm with four existing SFS algorithms, grafting [9], information-investing [6], fast-OSFS [5] and DIA-RED [11]. In the grafting algorithm, the multi-layer perceptron (MLP) is adopted as learning model and the λ parameters are chosen using fivefold 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 $W_\Delta = 0.5$. For the fast-OSFS algorithm, the independence tests are G^2 tests for all fully categorical or integer datasets and Fishers z-tests for all datasets containing real-valued features. For both tests, the statistical significance level (α) is set to be 0.05. DIA-RED is implemented using the combination entropy [38] and the size of incremental attribute set (SIA) is set to be 1. In the proposed SFS-RS algorithm, the measure proposed in [15] is used as dependency function.

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

Table 3 The selected subsets sizes using SFS-RS with Different dependency settings

Dataset	γ	ρ	v		τ		$\mathcal{N}$	ϕ
			$\beta = 0.1$	$\beta = 0.2$	$\alpha = 0.85$	$\alpha = 0.9$		
dorothea	6	6	5	–	–	–	–	–
arcane	4	4	6	7	–	–	–	–
dexter	9	9	–	9	–	–	–	–
madelon	4	5	5	5	–	–	–	–
arrhythmia	10	12	12	17	12	15	12	12
mf	11	11	12	12	–	–	–	–
tm1	6	7	8	10	–	–	–	–
tm2	7	7	12	8	–	–	–	–
tm3	7	8	7	9	8	9	9	9
ionosphere	5	6	5	7	6	7	7	6
wine	3	3	3	4	3	4	4	4
credit	4	3	5	6	6	6	4	3

Table 4 Running times of SFS-RS with different dependency settings

Dataset	γ	ρ	v		τ		$\mathcal{N}$	ϕ
			$\beta = 0.1$	$\beta = 0.2$	$\alpha = 0.85$	$\alpha = 0.9$		
dorothea	694.7	727.3	721.0	–	–	–	–	–
arcane	86.4	91.9	117.6	141.3	–	–	–	–
dexter	593.7	602.6	–	852.9	–	–	–	–
madelon	62.8	69.2	64.3	61.9	–	–	–	–
arrhythmia	21.0	25.5	39.5	63.1	766.1	1125.3	955.0	1002.9
mf	301.6	297.4	289.2	300.8	–	–	–	–
tm1	20.0	24.4	37.7	43.6	–	–	–	–
tm2	23.1	22.8	51.0	30.7	–	–	–	–
tm3	16.5	18.3	18.1	18.9	1281.8	1322.1	1183.9	2719.4
ionosphere	1.5	2.4	1.8	2.0	233.2	238.1	204.8	409.5
wine	0.3	0.3	0.3	0.5	47.1	44.8	42.5	76.1
credit	3.7	3.9	3.4	4.8	987.2	937.2	763.1	1622.0

Table 5 Classification accuracies of SFS-RS with different dependency settings

Dataset	γ	ρ	v		τ		$\mathcal{N}$	ϕ
			$\beta = 0.1$	$\beta = 0.2$	$\alpha = 0.85$	$\alpha = 0.9$		
dorothea	93.6	92.6	71.2	–	–	–	–	–
arcane	87.1	83.3	69.6	82.4	–	–	–	–
dexter	90.8	92.0	–	92.0	–	–	–	–
madelon	72.8	74.2	74.2	74.2	–	–	–	–
arrhythmia	96.1	98.3	82.5	96.8	93.2	62.8	98.3	98.3
mf	82.7	91.0	42.1	90.6	–	–	–	–
tm 1	98.3	99.4	93.2	93.2	–	–	–	–
tm 2	92.9	97.5	88.0	91.5	–	–	–	–
tm3	58.0	85.0	54.0	66.0	87.0	88.0	87.0	88.0
ionosphere	49.3	88.1	53.2	91.0	86.9	74.6	87.8	87.8
wine	82.1	89.2	89.2	80.7	89.2	80.7	80.7	80.7
credit	55.3	72.1	64.7	47.0	76.3	68.1	55.3	80.2

Table 6 reports the compactness of the selected subsets using the four algorithms. As it can be seen, the proposed algorithm selects fewer features than the other four algorithms. For datasets with larger feature sets (*dorothea*, *arcane*, and *dexter*), grafting and information-investing found subsets which are considerably large. This can be attributed to the nesting effect of the two algorithms. Moreover, the fast-OSFS algorithm failed to select a feature subset for five datasets, *arcene*, *tm3*, *ionosphere*, *wine*, and *credit*. This is related to the failure of the conditional independence tests to be applied to limited number of training instances. DIA-RED lost the comparison for all the tests. This algorithm failed to finish in a reasonable time for *dorothea*, *arcane*, *dexter*, *mf*, and *tm3*. This can be related to the fact that DIA-RED does not implement an effective redundant attribute elimination mechanism, and therefore the selected subsets are large during features streaming, which causes ineffective partitioning steps in calculating the rough sets approximations, and therefore, this algorithm is not time efficient.

The running time results are reported in Table 7. We see that the SFS-RS is superior for six datasets, *dorothea*, *arcene*, *dexter*, *madelon*, *tm2*, and *tm3*. The running times of fast-OSFS are comparable with the proposed method. Another important result is that the grafting, information-investing and DIA-RED algorithms are not time-efficient for datasets with large feature space.

The classification results, reported in Table 8, show that the proposed algorithm performs very well and shows increase in classification accuracies for most of the tests. Although grafting won the test for the *tm1*, SFS-RS shows an increase of up to 60 % for *dorothea* dataset. Compared with information-investing, our proposed algorithm is superior in all tests, except the *credit* dataset. Comparing with fast-OSFS and DIA-RED, the proposed algorithm is superior in all tests.

Figure 3 represents the classification accuracy of selected subsets during features streaming, for the three higher dimensional datasets *dorothea*, *arcane*, and *dexer*. A general conclusion from this figure is that the proposed

Table 6 Selected subsets size comparison of the five SFS algorithms

Dataset	SFS-RS	Grafting	Information-investing	Fast-OSFS	DIA-RED
dorothea	**6**	113	97	9	–
arcane	**4**	13	15	–	–
dexter	**9**	14	18	11	–
madelon	5	9	6	4	61
arrhythmia	12	12	10	7	46
mf	**11**	18	12	12	–
tm 1	7	7	7	7	29
tm 2	7	7	8	7	37
tm3	8	7	8	–	–
ionosphere	6	6	6	–	15
wine	3	3	3	–	6
credit	**3**	5	6	–	6

Table 7 Running times comparison of the five SFS algorithms

Dataset	SFS-RS	Grafting	Information-investing	Fast-OSFS	DIA-RED
dorothea	**727.3**	14576.02	13829.72	940.18	–
arcane	**91.9**	398.73	281.88	–	–
dexter	**602.6**	1091.77	1886.63	620.19	–
madelon	**69.2**	119.76	98.01	80.11	11283
arrhythmia	25.5	43.12	29.23	17.16	6392
mf	297.4	432.23	296.23	289.33	–
tm 1	24.4	21.72	23.32	17.87	7783
tm 2	**22.8**	35.68	38.76	30.88	8035
tm3	**18.3**	56.82	118.66	–	–
ionosphere	**2.4**	4.1	4.3	–	8.9
wine	0.3	0.3	0.3	–	1.9
credit	3.9	4.0	3.9	–	4.9

Table 8 Classification accuracy comparison of the five SFS algorithms

Dataset	SFS-RS	Grafting	Information-investing	Fast-OSFS	DIA-RED
dorothea	**92.6**	41.4	42.6	89.5	–
arcane	**83.3**	69.3	53.8	–	–
dexter	**92.0**	77.3	48.0	85.9	–
madelon	74.2	29.6	37.2	78.7	62.0
arrhythmia	**98.3**	81.9	87.5	96.4	32.2
mf	**91.0**	63.6	85.1	83.2	–
tm 1	99.4	100	82.8	89.9	29.4
tm 2	**97.5**	96.5	93.0	82.8	41.3
tm3	**85.0**	65.0	65.0	–	–
ionosphere	**88.1**	87.1	81.4	–	56.3
wine	**89.2**	77.9	77.9	–	58.9
credit	72.1	72.1	78.0	–	66.1

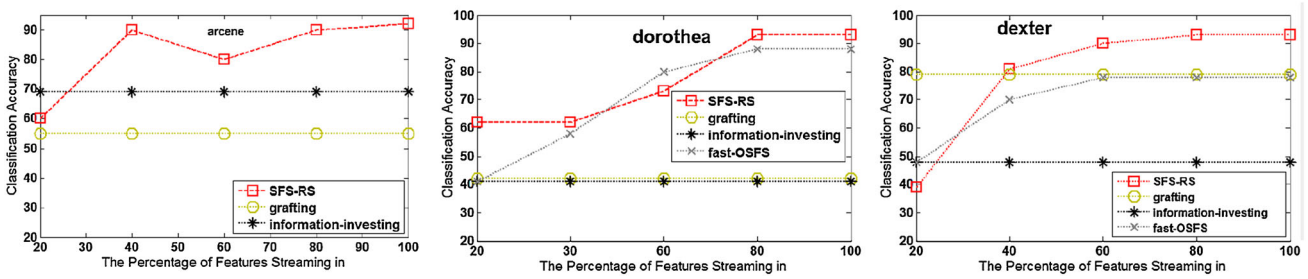

Fig. 3 Classification results of selected subsets during features streaming

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.

5 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 SFS problem from the rough sets (RS) perspective. The main motivation was that RS-based data mining do not require any domain knowledge other than the given dataset. A new SFS algorithm, called SFS-RS, is proposed. This algorithm adopts the feature significance concept to eliminate features which have no influence in deciding output feature.

To show the efficiency and accuracy of the proposed algorithm, it was compared with grafting, information-investing, fast-OSFS, and DIA-RED algorithms. Twelve high dimensional datasets were used for comparisons, and their features were considered one by one to simulate the true SF scenarios. The experiments demonstrated that the proposed algorithm achieves better results than existing SFS algorithms, for all evaluation terms.

References

1. Bishop CM (2006) Pattern recognition and machine learning (information science and statistics). Springer-Verlag New York Inc., Secaucus
2. Theodoridis S, Koutroumbas K (2009) Pattern recognition. Academic Press, Cambridge
3. Guyon I, Elliseff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
4. Wang J, Zhao P, Hoi S, Jin R (2014) Online feature selection and its applications. IEEE Trans Knowl Data Eng 26(3):698–710
5. Wu X, Yu K, Ding W, Wang H, Zhu X (2013) Online feature selection with streaming features. IEEE Trans Pattern Anal Mach Intell 35:1178–1192
6. Ungar L, Zhou J, Foster D, Stine B (2005) Streaming feature selection using IIC. In: Proceedings of the 10th International Conference on Articial Intelligence and Statistics
7. He YL, Liu JNK, Hu YH, Wang XZ (2015) OWA operator based link prediction ensemble for social network. Expert Syst Appl 42(1):21–50
8. Perkins S, Lacker K, Theiler J (2003) Grafting: fast, incremental feature selection by gradient descent in function space. J Mach Learn Res 3:1333–1356
9. Perkins S, Theiler J (2003) Online feature selection using grafting. In: International Conference on Machine Learning. ACM Press, pp 592–599

10. Pudil P, Novoviov J, Kittler J (1994) Floating search methods in feature selection. Pattern Recogn Lett 15(11):1119–1125
11. Wang F, Liang J, Qian Y (2013) Attribute reduction: a dimension incremental strategy. Knowl Based Sys 39:95–108
12. Hedar AR, Wang J, Fukushima M (2008) Tabu search for attribute reduction in rough set theory. Soft Comput 12(9):909–918
13. Li HR, Zhang WX (2005) Applying indiscernibility attribute sets to knowledge reduction. In: AI 2005: advances in artificial intelligence, vol 3809. Springer, Berlin, Heidelberg, pp 816–821. doi:10.1007/11589990_87
14. Li K, Liu YS (2002) Rough set based attribute reduction approach in data mining. In: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 1, pp 60–63
15. Parthalain N, Shen Q, Jensen R (2010) A distance measure approach to exploring the rough set boundary region for attribute reduction. IEEE Trans Knowl Data Eng 22(3):305–317
16. Jensen R, Tuson A, Shen Q (2014) Finding rough and fuzzy-rough set reducts with SAT. Inf Sci 255:100–120
17. Weihua X, Yuan L, Xiuwu L (2012) Approaches to attribute reductions based on rough set and matrix computation in inconsistent ordered information systems. Knowl Based Syst 27:78–91
18. Wang XZ (2015) Learning from big data with uncertainty–editorial. J Intell Fuzzy Sys 28(5):2329–2330
19. Wang XZ, Ashfag RAR, Fu AM (2015) Fuzziness based sample categorization for classifier performance improvement. J Intell Fuzzy Sys 29(3):1185–1196
20. He YL, Wang XZ, Huang JZX (2016) Fuzzy nonlinear regression analysis using a random weight network. Inf Sci 364–365: 222–240
21. Pawlak Z (1982) Rough sets. Int J Comput Inform Sci 11(5):341–356
22. Wentao L, Weihua X (2015) Double-quantitative decision-theoretic rough set. Inf Sci 316:54–67
23. Eskandari S, Javidi MM (2016) Online streaming feature selection using rough sets. Int J Approx Reason 69:35–57
24. Swiniarski RW, Skowron A (2003) Rough set methods in feature selection and recognition. Pattern Recogn Lett 24(6):833–849
25. Jensen R, Shen Q (2001) 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
26. Jensen R, Shen Q (2004) Semantics-preserving dimensionality reduction: rough and fuzzy-rough based approaches. IEEE Trans Knowl Data Eng 16(16):1457–1471
27. Ziarko W (1993) Variable precision rough set model. J Comput Syst Sci 46(1):39–59
28. Skowron A, Stepaniuk J (1996) Tolerance approximation spaces. Fundam Inform 27(2–3):245–253
29. Dubois D, Prade H (1992) Putting rough sets and fuzzy sets together. In: Słowinskí R (ed) Intelligent decision support. Theory and decision library, vol 11. Springer, Netherlands, pp 203–232
30. Yong L, Wenliang H, Yunliang J, Zhiyong Z (2014) Quick attribute reduct algorithm for neighborhood rough set model. Inf Sci 271:65–81
31. Kumar SU, Inbarani HH (2015) A novel neighborhood rough set based classification approach for medical diagnosis. Proc Comput Sci 47:351–359
32. Hu Q, Yu D, Liu J, Wu C (2008) Neighborhood rough set based heterogeneous feature subset selection. Inf Sci 178(18):3577–3594
33. Ashfaq RAR, Wang XZ, Huang JZX, Abbas H, He YL (2016) Fuzziness based semi-supervised learning approach for intrusion detection system. Inf Sci. doi:10.1016/j.ins.2016.04.019 **(in press)**
34. Clopinet, Feature Selection Challenge, NIPS (2003). http://clopinet.com/isabelle/Projects/NIPS2003/. Accessed 06 March 2015
35. Blake C, Merz CJ (1998) UCI repository of machine learning databases. http://www.ics.uci.edu/mlearn/MLRepository.html. Accessed 06 March 2015
36. Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco
37. Chang CC, Lin CJ (2011) Libsvm: a library for support vector machines. ACM Trans Intell Sys Technol 2(3):1–27
38. Qian Y, Liang J (2008) Combination entropy and combination granulation in rough set theory. Int J Uncertain Fuzziness Knowl Based Sys 16(2):179–193