Artificial Intelligence Review (2023) 56:S1011–S1062
https://doi.org/10.1007/s10462-023-10546-9

Feature subset selection for data and feature streams: a review

Carlos Villa-Blanco[1] · **Concha Bielza**[1] · **Pedro Larrañaga**[1]

Accepted: 1 January 2022 / Published online: 13 July 2023

Abstract
Real-world problems are commonly characterized by a high feature dimensionality, which hinders the modelling and descriptive analysis of the data. However, some of these data may be irrelevant or redundant for the learning process. Different approaches can be used to reduce this information, improving not only the speed of building models but also their performance and interpretability. In this review, we focus on feature subset selection (FSS) techniques, which select a subset of the original feature set without making any transformation on the attributes. Traditional batch FSS algorithms may not be adequate to efficiently handle large volumes of data, either because memory problems arise or data are received in a sequential manner. Thus, this article aims to survey the state of the art of incremental FSS algorithms, which can perform more efficiently under these circumstances. Different strategies are described, such as incrementally updating feature weights, applying information theory or using rough set-based FSS, as well as multiple supervised and unsupervised learning tasks where the application of FSS is interesting.

Keywords Data streams · Feature streams · Dynamic environments · Feature subset selection · Supervised classification · Clustering

1 Introduction

Feature subset selection (FSS) is the task of choosing a subset of features, also known as independent/predictive variables/attributes, from a complete dataset, with the objective of improving the efficiency, precision and interpretability of built models. This can be achieved by using fewer variables since FSS seeks to discard irrelevant and redundant features that may be confusing and harmful for learning algorithms. The FSS task described

✉ Carlos Villa-Blanco
carlos.villa@upm.es

Concha Bielza
mcbielza@fi.upm.es

Pedro Larrañaga
pedro.larranaga@fi.upm.es

[1] Computational Intelligence Group, Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Boadilla del Monte, 28660 Madrid, Spain

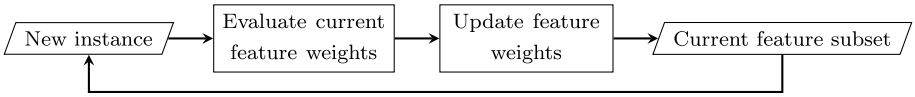


Fig. 1 Example of a framework of online FSS algorithms for data streams

in this article should not be confused with feature extraction, which constructs new variables from the available ones to obtain a lower-dimensional (albeit less intuitive) feature space.

This review focuses on *incremental* FSS algorithms, i.e., approaches used in environments where new training instances, features or both are received progressively over time and, therefore, the subset of candidate features should be updated dynamically. Traditional batch approaches cannot be directly applied in a streaming context, implying that models have to be retrained when new data are received, a solution that may be inefficient and poorly scalable when processing big data (Wu et al. 2017). In addition, incremental algorithms are not only useful when the entire set of features or instances (examples) are not available beforehand, but they can also be considered in the preprocessing step when all the data are available, but computational resources are limited (Jing et al. 2018).

Proposals that incrementally work with information flows are known by several terms, such as online or incremental. There is no well-established definition for these concepts and they are used interchangeably by some authors. However, we believe it is appropriate to consider online learning as a subtype of incremental learning, which imposes stricter time and space requirements and should be able to work endlessly in a streaming environment (Gama et al. 2014; Losing et al. 2018). It can be seen that the definition of an algorithm as online is truly dependent on its complexity when addressing real-time problems. An FSS algorithm could be adequately efficient to handle a certain online problem, given solutions at the expected time, but it may not be appropriate in a more demanding environment. If we focus on the complexity of the algorithm to define it as online or incremental, the definition would depend on the characteristics of the problem being addressed. Therefore, we concluded that online learning literature focuses more on defining algorithms that can be applied to real-time problems, while incremental algorithms simply seek to improve the efficiency obtained by batch proposals. In return, online algorithms, at least in the case of receiving new instances, may find their effectiveness reduced as the model complexity is additionally bounded (Losing et al. 2018). However, this definition does not perfectly fit when new features are received. If we face a theoretically infinite stream of features, where an infinite number of them could be considered relevant and non-redundant, it is inevitable that a memory problem will appear at some point. Nevertheless, algorithms that could face this problem are considered to be online algorithms in the literature. This article reviews algorithms defined as online or incremental by their authors that can be applied in a context where new data or features can be received, i.e., on data streams, feature streams or a combination of both. The objective is not to present an unequivocal way of defining an algorithm as online or incremental but to study diverse proposals that are capable of incrementally updating a current selection of variables with the arrival of new information.

Dynamic data can be categorized into three main classes: *data streams*, *feature streams* and a hybrid known as *data and feature stream*. When working with data streams, new instances arrive over time, while the feature set remains fixed. Thus, the set of selected features and the models should be adapted according to the new data. Figure 1 shows a framework commonly followed by online FSS algorithms when working with data streams,

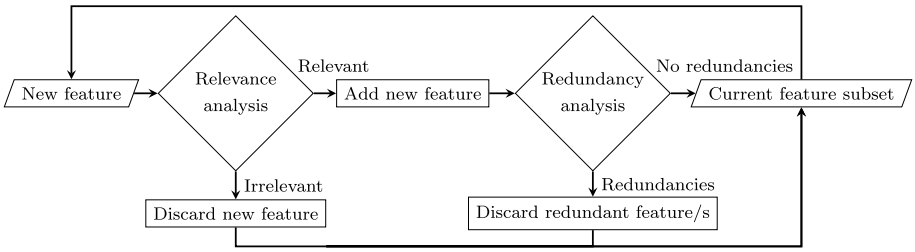


Fig. 2 General framework of FSS algorithms for feature streams

where a set of feature weights is dynamically updated according to certain rules. Given the arrival of a new instance (or instance group), the applied rules take into account, for example, whether a misclassification occurs or if the value of a loss function is above a certain threshold. If we consider other algorithms commonly defined as incremental, it is common to use temporal windows in conjunction with information theory measures, such as mutual information, to evaluate the feature importance. The appearance of new instances represents several challenges, such as model accuracy reduction due to changes in the underlying data relationships. This problem is known as *concept drift*, one of the main obstacles when considering data streams. There are real-world problems that involve handling data streams in very diverse areas, such as spam detectors (Wang et al. 2014), personalized medicine (Swan 2012) or analysis of currency exchange rates and demographic data (Yi et al. 2000).

Unlike data streams, feature streams are characterized by a fixed set of instances, but the feature set evolves over time; thus, the objective is to maintain a feature subset with the most relevant features that have arrived so far, avoiding redundancies. The general approach for incremental/online FSS on feature streams includes a relevance and redundancy analysis, as shown in Fig. 2, to assess whether a new variable (or variables) is relevant and, in a positive case, to check for redundancies caused by its inclusion in the subset of selected variables. This framework varies since some approaches, for example, do not include a redundancy analysis or require additional steps (see, for example, Sect. 4.2). Feature streams may be useful when (1) the generation of all the features is expensive and it is not feasible to wait for all of them to be generated, as could occur in some scientific experiments, or (2) when we do not have the means to obtain all possible features at the moment since, for example, they could be obtained from a social network and be based on current news. We can find several examples of FSS applications over feature streams, such as in medical learning problems (Wang et al. 2017), statistical relational learning (Zhou et al. 2005), texture-based image segmentation (Perkins and Theiler 2003), bioinformatics experiments (Wang et al. 2014), edge detection in grayscale images (Glocer et al. 2005) and analysis of social network data (Yu et al. 2016a).

Finally, a hybrid type of stream called a data and feature stream, which implies the appearance of both new instances and features over time, will be discussed.

FSS techniques are commonly categorized into three types depending on their relationship with the model construction: the *filter*, *embedded* and *wrapper* methods (Saeys et al. 2007; Guyon et al. 2008). A filter algorithm uses only the training data and a metric to score each feature or subset of features, so the result is independent of learning models. Embedded techniques include the feature selection process as a part of the model definition, i.e., embedded in the modelling algorithm. Wrapper algorithms evaluate different feature subsets by training a specific model for each of them and report the feature

combination whose model achieves the best testing performance (e.g., accuracy). The creation of several models every time new data are available may be computationally expensive and incompatible with the interest of online algorithms to evaluate new data, ideally, in real time. This scenario is even worse if new features are received since, without considering the use of any heuristic search, there would be $2^d - 1$ (for d features) possible feature combinations to evaluate and, therefore, models to build. We believe this is the reason why wrapper algorithms do not receive much attention for incremental learning, especially when considering online scenarios with more demanding time and memory constraints.

This article is motivated by the fact that reviews on incremental/online FSS are scarce, so that the vast majority of studies focus primarily on batch techniques (Pereira et al. 2018; Urbanowicz et al. 2018; Bolón-Canedo and Alonso-Betanzos 2019; Venkatesh and Anuradha 2019; Zhang et al. 2019; Solorio-Fernández et al. 2020; Pintas et al. 2021). However, the increased interest in algorithms with incremental mechanisms requires an in-depth study of the state of the art on this topic. Recent publications considering the streaming scenario only explain a really small and not very diverse set of methods compared to the number of existing contributions in this field (Hu et al. 2016; Tommasel and Godoy 2016; Li et al. 2017; Cai et al. 2018; Ma et al. 2018; Somasundaram and Mylsamy 2018; AlNuaimi et al. 2020). Most of the algorithms studied in these publications were proposed several years ago, so it is also necessary to contemplate new proposals that can handle the performance demands and learning problems that we may face in increasingly massive datasets and faster processes. This review article aims to provide a big picture of incremental FSS methods, presenting the reader with a detailed study of proposals with different characteristics and for a wide variety of learning problems. In addition, this comprehensive analysis has allowed us to detect an interesting variety of open issues in several areas of study. Nevertheless, this article not only reviews the most recent and prominent algorithms, but also clarifies and structures an area of study in which there is confusion both in the acronyms and definitions used and in the categorization of some proposals. In summary, the main objectives of this work are the following:

- Formal definition and introduction of taxonomies for the FSS problem, with a special interest in its application to data streams, feature streams or a combination of both.
- Comprehensive and structured study of a wide variety of incremental/online FSS algorithms for data and/or feature streams.
- Comparison of the studied algorithms, describing in detail their functioning and offering alternatives for some of their shortcomings.
- Study of an extensive variety of learning problems where incremental/online FSS is interesting to be applied. These include, among others, supervised (binary, multi-class and multi-label), unsupervised, multi-task, multi-view or ensemble learning, discrete and continuous features, receiving instances/features individually or in groups, or rough set-based approaches.

The remainder of this paper is organized as follows. In Sect. 2, the FSS problem over streams of data or features is described and categorized, defining the nomenclature that will be used throughout this document. In Sect. 3, several FSS algorithms that can be used with data streams are introduced and categorized as supervised or unsupervised approaches. In Sect. 4, FSS algorithms for feature streams are described and categorized into those that process new features individually or by groups. In turn, these two categories are divided into supervised and unsupervised algorithms. In Sect. 5, the incremental incorporation of both new instances and features is analyzed, and the few proposals that exist are studied. In

Sect. 6, future work on incremental/online FSS is outlined. Finally, in Sect. 7, some conclusions achieved during the realization of this review are presented.

2 Problem statement

This section defines the mathematical notation that will be used throughout the paper, including an explanation of FSS, data streams, feature streams and the rough set theory, all illustrated with examples.

2.1 Feature subset selection

Let $\mathcal{D} = \{\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i), 1 \leq i \leq n\}$ be a static dataset, where $\mathbf{z}_i$ denotes the training instance i with a vector of values $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})$ of a fixed dimension d and a class vector $\mathbf{y}_i = (y_{i1}, \ldots, y_{il})$, where l is the number of class variables. Thus, $l = 1$ is a one-dimensional supervised classification problem and $l > 1$ is a multi-dimensional one. Each class variable Y_j has k_j distinct values, so if $k_j = 2$, $\forall j = 1, \ldots, l$, we have a binary ($l = 1$) or multi-label ($l > 1$) problem, while if $k_j > 2$ for some j, it can be a multi-class ($l = 1$) or multi-dimensional ($l > 1$) problem. Given that our set of features is represented by $\mathcal{X} = \{X_1, \ldots, X_d\}$, so value x_{ij} belongs to feature X_j, and our set of class variables by $\mathcal{Y} = \{Y_1, \ldots, Y_l\}$, so y_{ij} belongs to class variable Y_j, the feature selection task selects a subset of features $\mathcal{S} \subseteq \mathcal{X}$ that can be used to obtain a mapping function f from $\mathcal{S}$ to $\mathcal{Y}$ that is as good as possible for a certain criterion. This subset of features $\mathcal{S}$ should contain the most relevant features while avoiding redundancies.

If the dataset is unsupervised, it may not be clear how to evaluate the importance of a feature since there are no class variables, $\mathcal{Y} = \emptyset$, that could be used as a reference to define their relevance. The ideal aim of unsupervised FSS is therefore to preserve the most important characteristics of the data in a reduced feature space that contains the most discriminative features for the task to be performed. The following section offers an in-depth view of the definition of relevant and redundant features in supervised and unsupervised learning problems.

2.2 Relevant and redundant features

There is no unique way to define relevant and redundant features: the definition is dependent on the task for which the feature selection is made. For example, the authors in (Wang et al. 2014; Yang et al. 2013) learn sparse models, i.e., models where parameters of the least important features shrink to zero, by imposing certain constraints, such as L1-norm regularization, whereas the authors in (Yu et al. 2014; Domingos and Hulten 2000; Liang et al. 2014) explicitly define the goodness of the features based on measures such as entropy, mutual information or Gini index. By contrast, wrapper algorithms could simply report a feature subset based on the model that achieves the highest accuracy or area under the ROC curve.

Formally, in a probabilistic context, feature relevance is commonly defined as in John et al. (1994), which states that, in supervised learning, relevant features can be divided into two groups: strongly relevant and weakly relevant features. Strongly relevant features are indispensable for the final subset of selected features since their removal would imply a

loss of prediction power. Given a feature X_i and a set of features $\mathcal{T} = \mathcal{X} \backslash \{X_i\}$, X_i is strongly relevant to a class variable Y_j, iff:

$$p(Y_j|X_i, \mathcal{T}) \neq p(Y_j|\mathcal{T}).$$

A weakly relevant feature may contribute to increasing the prediction power of a model, which depends on the other features that are available. Then, a weakly relevant feature can become strongly relevant by the removal of other features. X_i is considered to be weakly relevant iff it is not strongly relevant and there exists $\mathcal{T}' \subset \mathcal{T}$ such that:

$$p(Y_j|X_i, \mathcal{T}') \neq p(Y_j|\mathcal{T}').$$

If a feature is not strongly or weakly relevant, then it does not contribute to the prediction power, i.e., it is an irrelevant feature.

Defining only the most relevant features may not be sufficient to perform an effective FSS. A redundancy method that removes features that are highly correlated with other relevant ones and, therefore, do not provide extra information about classes is necessary. This task is usually performed by removing high correlations between features.

Following Yu and Liu (2004), for a class variable Y_j, a feature $X_i \in \mathcal{X}$ can be formally defined as redundant iff it is weakly relevant and has a Markov blanket $\mathcal{M}_i$ in $\mathcal{X}$ ($X_i \notin \mathcal{M}_i$), i.e., there is a feature subset that contains the information that X_i has about Y_j. $\mathcal{M}_i$ is a Markov blanket of X_i iff:

$$p(\mathcal{X} \setminus \{\mathcal{M}_i, X_i\}, Y_j|X_i, \mathcal{M}_i) = p(\mathcal{X} \setminus \{\mathcal{M}_i, X_i\}, Y_j|\mathcal{M}_i).$$

Notably, algorithms for data streams that are defined as online do not explicitly focus on identifying redundancies but expect that zero-valued feature weights belong to redundant and irrelevant features. Following the previous definition would be inefficient for the objectives of online learning since it would require analyzing several feature combinations every time new data are obtained. On the other hand, incremental algorithms for data streams and proposals for feature streams could consider explicitly defining whether a feature is redundant. In the case of proposals for feature streams that consider redundancy analysis, they may, for example, establish a limit to the size of the Markov blankets (Wu et al. 2013) or approximate them (Yu et al. 2014).

That said, Yu and Liu (2004) divide a set of features into four disjoint components: (1) irrelevant features, (2) weakly relevant and redundant features, (3) weakly relevant but non-redundant features and (4) strongly relevant features, defining the optimal subset of features as formed by (3) and (4).

The following example, inspired by Yu and Liu (2004), attempts to clarify the concepts of irrelevant, strongly/weakly relevant and redundant features.

Example 1 Given a set of binary features $\{X_1, X_2, X_3, X_4\}$, where $X_2 = \overline{X}_3$, and a class variable $Y = f(X_1, X_2)$ obtained with function $f(\cdot, \cdot)$, it is easy to see that the two possible optimal feature subsets are formed by $\{X_1, X_2\}$ or $\{X_1, X_3\}$. Then, X_1 is a strongly relevant feature, while X_2 and X_3 are weakly relevant. Only one of these latter two features should be removed since they are redundant. Feature X_4 is irrelevant for the computation of Y. ■

Despite the fact that most studies are focused on supervised problems, there is a wide range of areas that involve high-dimensional unlabelled data. In this situation, it is not trivial to select the most important features since we do not have the information provided by

Table 1 Example dataset of phishing websites

	LongURL	*LinksToPage*	*AgeDomain*	*IsReliable*
$\mathbf{z}_1$	0	1	1	0
$\mathbf{z}_2$	1	1	2	1
$\mathbf{z}_3$	1	0	3	1
$\mathbf{z}_4$	1	1	2	0
$\mathbf{z}_5$	0	1	1	0
$\mathbf{z}_6$	1	0	3	1

the class variables. A common solution is to extract information from the data that could be employed to generate a target variable. Then, the obtained pseudo-class labels can be used with a supervised FSS approach, such as obtaining sparse feature weights with a regularized regression (Huang et al. 2015). Another simpler approach, commented on in this article, is to use a similarity measure to obtain a predefined number of feature clusters and select a representative feature from each of them (Almusallam et al. 2018).

2.3 FSS on data streams

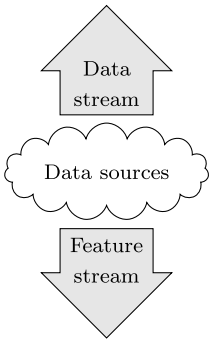
As explained in Sect. 1, there are certain problems where not all the training instances are available at the beginning but are obtained dynamically from a data stream $\mathcal{D} = lim_{T\to\infty} \bigcup_{t=1}^{T} \{\mathbf{z}_i^t = (\mathbf{x}_i^t, \mathbf{y}_i^t), 1 \leq i \leq n_t\} = lim_{T\to\infty} \bigcup_{t=1}^{T} \mathcal{D}^t$. Therefore, new training instances $\mathbf{z}_i^t$ can be received at each time t, and the number of time steps T could be unknown and theoretically infinite. It is even possible to encounter problems where the set of instances is reduced over time. The task of feature selection on data streams is to dynamically select a subset of features $\mathcal{S}^t$ at each time step t that contains most of the information provided by the complete feature set. Note that in the case of an unsupervised problem, a training instance would not have class variables, i.e., $\mathbf{z}_i^t = \mathbf{x}_i^t$.

Example 2 For the following examples, we use the dataset for fraudulent website analysis from Table 1. As shown at the top of Fig. 3, the instances of the dataset could have been sequentially received from a data stream until time $t = 3$. Then, at $t = 4$, a new website could be analyzed, and its information for the four available features would be included. These new data could be obtained from multiple kinds of sources, such as sensors, social networks and lab experiments. ■

2.3.1 Concept drift

The evolving environment under study implies that the underlying distribution of the data can change over time, which produces the appearance of concept drift. A concept at a time t is typically defined as the joint distribution between the feature space and, in the case of supervised learning, the target space, i.e., $p^t(\mathcal{X}, \mathcal{Y})$ (Gama et al. 2014). Thus, concept drift

	LongURL	*LinksToPage*	*AgeDomain*	*IsReliable*
$\mathbf{z}_1^1$	0	1	1	0
$\mathbf{z}_2^1$	1	1	2	1
$\mathbf{z}_3^2$	1	0	3	1
$\mathbf{z}_4^2$	1	1	2	0
$\mathbf{z}_5^3$	0	1	1	0
$\mathbf{z}_6^3$	1	0	3	1
$\mathbf{z}_7^4$	1	1	2	1



	LongURL$(t=1)$	*LinksToPage*$(t=1)$	*AgeDomain*$(t=1)$	*IsReliable*$(t=1)$	*DNSRecord*$(t=2)$	*StatisticalReport*$(t=3)$
$\mathbf{z}_1$	0	1	1	0	0	1
$\mathbf{z}_2$	1	1	2	1	0	1
$\mathbf{z}_3$	1	0	3	1	1	1
$\mathbf{z}_4$	1	1	2	0	1	0
$\mathbf{z}_5$	0	1	1	0	0	0
$\mathbf{z}_6$	1	0	3	1	1	1

Fig. 3 Example of data (top) and feature (bottom) streams of phishing websites

occurs between t and $t+1$ if the instances obtained from the data stream at those time instants are generated by different probability distributions:

$$p^t(\mathcal{X}, \mathcal{Y}) \neq p^{t+1}(\mathcal{X}, \mathcal{Y}).$$

There are two types of concept drift that are of special interest: *real concept drift* (*class drift*), which implies changes in the underlying relationships between features and class variables (supervised learning), i.e., $p^t(\mathcal{Y}|\mathcal{X}) \neq p^{t+1}(\mathcal{Y}|\mathcal{X})$, and *virtual concept drift* (*covariate drift*), which is related to changes in the distribution of features (supervised and unsupervised learning), i.e., $p^t(\mathcal{X}) \neq p^{t+1}(\mathcal{X})$.

The interest in detecting these distribution changes is that they may alter the subset of features that should be considered relevant. This is known as *feature drift* and occurs when the relevant subset $\mathcal{S}^t \subseteq \mathcal{X}$ for a certain task at time t differs from the subset obtained at another consecutive time instant $t+1$, i.e., $\mathcal{S}^t \neq \mathcal{S}^{t+1}$ (Nguyen et al. 2012).

Finally, the existence of other kinds of concept drift should be noted, for example, novel class appearance, which occurs due to the appearance of new classes. The reader is referred to (Webb et al. 2016; Barddal et al. 2017) for a more in-depth discussion of concept and feature drift.

2.4 FSS on feature streams

In the case of a feature stream, the number of instances will not increase, so n is a fixed known value. However, the set of features $\mathcal{X}$ is not fixed, but d is incremented (or even reduced) over time. When a new feature becomes available, all its values for each instance are received, and the subset of selected features $\mathcal{S}^t$ must be dynamically adapted to represent the most relevant features seen so far and avoid redundancies. Most of the existing approaches process new features individually, but it is also possible to receive features in groups. A group of features arriving at time t is represented as $\mathcal{G}^t = \{X_j^t, 1 \leq j \leq d_t\}$, where X_j^t is its j-th feature and d_t is the number of features that contains. Thus, in the limit, we would obtain a set of feature groups $\mathcal{G} = \bigcup_{t=1}^{\infty} \mathcal{G}^t$.

Example 3 Continuing with the dataset of Table 1, but in this case assuming all instances to be available in advance, new information about the websites could be obtained at times $t = 2$ and $t = 3$, as described at the bottom of Fig. 3. In this case, the new data at $t = 2$ is related to whether the DNS record is found, and the data at $t = 3$ is the result of a new statistical report. The data are included as new features that contain values for all the available instances. ■

In the feature stream context, the problem of concept drift does not arise. In two different time instants, the subset of relevant features may change, which is the definition of feature drift. However, this is caused by the appearance of new variables that make previous ones redundant. As the set of instances never varies, the distribution of the data does not change.

2.5 Rough set theory

When applying FSS algorithms, we are not restricted to a unique domain; we can handle very diverse data. That is why some studies seek approaches that do not need prior knowledge about the domain, except the given data. This is the case for incremental FSS algorithms under approaches such as *rough set theory*, a mathematical tool to express vagueness (due to lack of information) by a boundary region (Pawlak 1997) without needing any domain knowledge apart from the given data. Rough set theory enables working on problems where knowledge is incomplete, so it is ideal for the streaming environment under study since the complete dataset is unknown at the beginning of the process. Incremental FSS based on rough set theory, commonly known as incremental attribute reduction, is an important task for knowledge acquisition, and algorithms based on this theory are currently receiving considerable attention (Zhang et al. 2016; Xu et al. 2011). Thus, we believe they should be included in this article.

Rough set theory was originally proposed by Pawlak (1982), and it describes a subset of a universe using two subsets, the *lower and upper approximations*. This division allows hidden knowledge to be discovered in datasets[1] and expressed with decision rules. A dataset is defined as a tuple $\mathcal{D} = < \mathcal{U}, \mathcal{A} >$, where $\mathcal{U}$ is a set of objects or instances, known as the universe, and $\mathcal{A} = \{\mathcal{X} \cup \mathcal{Y}\}$ is a set of attributes, where $\mathcal{X}$ is the set of features and $\mathcal{Y}$ is the set of class variables, such that $\mathcal{X} \cap \mathcal{Y} = \emptyset$.

[1] Commonly known as an information system or decision information system if there are class variables.

Rough set theory used the indiscernibility relation between instances to divide them into disjoint sets of similar instances. This relationship is present when all instances of a set have the same values for all the studied attributes. Thus, each non-empty subset $\mathcal{B} \subseteq \mathcal{A}$ determines a $\mathcal{B}$-indiscernibility relation $IND(\mathcal{B}) = \{(\mathbf{z}_i, \mathbf{z}_j) \in \mathcal{U} \times \mathcal{U} \mid \mathbf{z}_i^{\downarrow\mathcal{B}} = \mathbf{z}_j^{\downarrow\mathcal{B}}\}$, where $\mathbf{z}_i^{\downarrow\mathcal{B}}$ represents the projection of instance $\mathbf{z}_i$ on attributes in $\mathcal{B}$. This indiscernibility relation partitions $\mathcal{U}$ into disjoint subsets of indistinguishable instances, which are known as equivalence classes or elementary sets. This partition is denoted as $\mathcal{U}/IND(\mathcal{B}) = \{\mathcal{E}_1, \ldots, \mathcal{E}_v\}$, where v is the number of equivalence classes. To represent the equivalence class with respect to $\mathcal{B}$ that contains a certain instance $\mathbf{z}_i$, we use the expression $[\mathbf{z}_i]_\mathcal{B}$, i.e., $[\mathbf{z}_i]_\mathcal{B} = \{\mathbf{z}_j \in \mathcal{U} \mid \mathbf{z}_i^{\downarrow\mathcal{B}} = \mathbf{z}_j^{\downarrow\mathcal{B}}\}$. Notably, equivalence classes formed only by features, i.e., $\mathcal{U}/IND(\mathcal{X})$, or by class variables, i.e., $\mathcal{U}/IND(\mathcal{Y})$, are known as condition and decision classes, respectively.

Given any subset $\mathcal{H} \subseteq \mathcal{U}$, rough set theory seeks to approximate $\mathcal{H}$ using a lower and upper bound, which are defined with the equivalence classes induced by a feature set $\mathcal{B}$ over $\mathcal{U}$. These sets are the $\mathcal{B}$-lower approximation of $\mathcal{H}$, denoted as $\underline{\mathcal{B}}\mathcal{H}$, and the $\mathcal{B}$-upper approximation of $\mathcal{H}$, denoted as $\overline{\mathcal{B}}\mathcal{H}$. The $\mathcal{B}$-lower approximation contains all the instances that certainly belong to $\mathcal{H}$ with the information given by $\mathcal{B}$, i.e., $\underline{\mathcal{B}}\mathcal{H} = \{\mathbf{z} \mid [\mathbf{z}]_\mathcal{B} \subseteq \mathcal{H}\}$, while the $\mathcal{B}$-upper approximation includes those instances that can possibly belong to $\mathcal{H}$, i.e., $\overline{\mathcal{B}}\mathcal{H} = \{\mathbf{z} \mid [\mathbf{z}]_\mathcal{B} \cap \mathcal{H} \neq \emptyset\}$. Finally, a rough set of $\mathcal{H}$ with respect to $\mathcal{B}$ is defined as the tuple formed by the $\mathcal{B}$-lower and $\mathcal{B}$-upper approximations, i.e., $< \underline{\mathcal{B}}\mathcal{H}, \overline{\mathcal{B}}\mathcal{H} >$. These approximations divide the universe into three disjoint regions: (1) the positive region $POS_\mathcal{B}^\mathcal{U}(\mathcal{H}) = \underline{\mathcal{B}}\mathcal{H}$, which contains all the instances from $\mathcal{U}$ that can be classified as certainly belonging to $\mathcal{H}$, (2) the negative region $NEG_\mathcal{B}^\mathcal{U}(\mathcal{H}) = \mathcal{U} \backslash \overline{\mathcal{B}}\mathcal{H}$, which represents the instances that for sure do not belong to $\mathcal{H}$, and (3) the boundary region $BND_\mathcal{B}^\mathcal{U}(\mathcal{H}) = \overline{\mathcal{B}}\mathcal{H} \backslash \underline{\mathcal{B}}\mathcal{H}$, which consists of those instances that cannot be classified as belonging or not to $\mathcal{H}$ due to a lack of knowledge. If $BND_\mathcal{B}^\mathcal{U}(\mathcal{H}) = \emptyset$, then $\mathcal{H}$ is considered crisp (exact) since it is possible to define all instances as members or not of $\mathcal{H}$. Otherwise, the set is rough (inexact) with respect to $\mathcal{B}$, i.e., we cannot define the set precisely with the available knowledge.

For a certain dataset with class variables $\mathcal{Y}$ and given $\mathcal{H} \in \mathcal{U}/IND(\mathcal{Y})$, the positive, negative and boundary regions for a feature set $\mathcal{B}$ given $\mathcal{Y}$ are: $POS_\mathcal{B}^\mathcal{U}(\mathcal{Y}) = \bigcup POS_\mathcal{B}^\mathcal{U}(\mathcal{H})$, $NEG_\mathcal{B}^\mathcal{U}(\mathcal{Y}) = \bigcup NEG_\mathcal{B}^\mathcal{U}(\mathcal{H})$ and $BND_\mathcal{B}^\mathcal{U}(\mathcal{Y}) = \bigcup BND_\mathcal{B}^\mathcal{U}(\mathcal{H})$, respectively. These regions are obtained by joining the resulting partitions of the universe for each of the decision classes induced by $\mathcal{Y}$.

As explained before, not all the available features may be necessary. Thus, attribute reduction seeks a minimal feature subset (or all possible subsets) that is sufficient to characterize the knowledge of a dataset, i.e., it discards all unnecessary features while preserving certain properties of the original dataset. This subset is called the minimal reduct, which could be required, for example, to maintain the same positive region as the one obtained with the original feature set. To respect the notation used so far, the reducts are denoted by $\mathcal{S}$. As we are working in a dynamic environment, the reducts should be updated according to currently available data.

Example 4 Given the dataset of Table 1 with features $\mathcal{B} = \{LongURL, LinksToPage, AgeDomain\}$ and $\mathcal{U} = \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4, \mathbf{z}_5, \mathbf{z}_6\}$, it is possible to find that instances such as $\mathbf{z}_1$ and $\mathbf{z}_5$ are indiscernible with respect to $\mathcal{B}$, so equivalence classes $\{\mathbf{z}_1, \mathbf{z}_5\}$, $\{\mathbf{z}_2, \mathbf{z}_4\}$ and $\{\mathbf{z}_3, \mathbf{z}_6\}$ are obtained. If the target set to approximate is

Table 2 Summary of the main notation used throughout the article

Symbol	*Description*
$\mathcal{X} = \{X_1, \ldots, X_d\}$	Set of features
$\mathcal{Y} = \{Y_1, \ldots, Y_l\}$	Set of class variables
$\mathcal{S}^t \subseteq \mathcal{X}$	Subset of selected features or reduct at time t
$\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$	i-th instance
$\mathbf{x}_i = (x_{i1}, \ldots, x_{id})$	Feature values for the i-th instance
x_{ij}	Value of feature X_j for the i-th instance
$\mathbf{y}_i = (y_{i1}, \ldots, y_{il})$	Class variable values for the i-th instance
y_{ij}	Value of class variable Y_j for the i-th instance
k_j	Number of distinct values of class variable Y_j
$\mathcal{G}^t = \{X_j^t, 1 \leq j \leq d_t\}$	Group of d_t features arriving at time t
$\mathbf{w}$ ($\mathbf{W}$)	Bold lower (upper) case letters denote vectors (matrices)
$\mathbf{w}_{i\bullet}$ ($\mathbf{w}_{\bullet i}$)	i-th row (column) of matrix $\mathbf{W}$
$\mathcal{U}$	Universe (set of instances in rough set theory)
$\mathcal{U}/IND(\mathcal{X}) = \{\mathcal{E}_1, \ldots, \mathcal{E}_v\}$	Indiscernibility relation given a feature set $\mathcal{X}$
$\mathcal{E}$	Equivalence class
$POS_{\mathcal{X}}^{\mathcal{U}}(Y)$	Positive region for a feature set $\mathcal{X}$ given class variables $\mathcal{Y}$
$NEG_{\mathcal{X}}^{\mathcal{U}}(Y)$	Negative region for a feature set $\mathcal{X}$ given class variables $\mathcal{Y}$
$BND_{\mathcal{X}}^{\mathcal{U}}(Y)$	Boundary region for a feature set $\mathcal{X}$ given class variables $\mathcal{Y}$

$\mathcal{H} = \{\mathbf{z} | \mathcal{Y}(\mathbf{z}) = 1\}$, i.e., $\mathcal{H} = \{\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_6\}$, where $\mathcal{Y} = \{IsReliable\}$, we can determine that $\underline{\mathcal{B}}\mathcal{H} = \{\mathbf{z}_3, \mathbf{z}_6\}$ and $\overline{\mathcal{B}}\mathcal{H} = \{\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_4, \mathbf{z}_6\}$. Thus, $POS_{\mathcal{B}}^{\mathcal{U}}(\mathcal{H}) = \{\mathbf{z}_3, \mathbf{z}_6\}$, $NEG_{\mathcal{B}}^{\mathcal{U}}(\mathcal{H}) = \{\mathbf{z}_1, \mathbf{z}_5\}$ and $BND_{\mathcal{B}}^{\mathcal{U}}(\mathcal{H}) = \{\mathbf{z}_2, \mathbf{z}_4\}$. As $BND_{\mathcal{B}}^{\mathcal{U}}(\mathcal{H}) \neq \emptyset$, set $\mathcal{H}$ is rough. If the values of $\mathbf{z}_4$ are changed to, for example, $LongURL(\mathbf{z}_4) = 0$, $LinksToPage(\mathbf{z}_4) = 1$ and $AgeDomain(\mathbf{z}_4) = 1$, the boundary region would be empty and $\mathcal{H}$ would be crisp. This is easy to see in our example since those changes would make $\underline{\mathcal{B}}\mathcal{H} = \overline{\mathcal{B}}\mathcal{H} = \{\mathbf{z}_2, \mathbf{z}_3, \mathbf{z}_6\}$, allowing us to undoubtedly classify all the instances as belonging or not to $\mathcal{H}$. Another way to achieve this could be including new features to $\mathcal{B}$.

Finally, whether we consider the original dataset or its version with $\mathbf{z}_4$ modified, set $\mathcal{B}$ can be further reduced to $\mathcal{S} = \{LongURL, LinksToPage\}$ without losing significant information since the same approximation sets are obtained. This set cannot be further reduced while maintaining the same approximations. Thus, it can be assured that it is a minimal reduct. ■

For the sake of clarity, Table 2 provides a summary of the main notation that is used in this work.

2.6 Classification of incremental FSS approaches

Figure 4 summarizes the classification criteria for the incremental FSS algorithms used throughout this review. Different algorithms for data streams and feature streams, which are classified depending on how the learning is performed (supervised or unsupervised), the number of available class variables and the arrival of new features (individual or

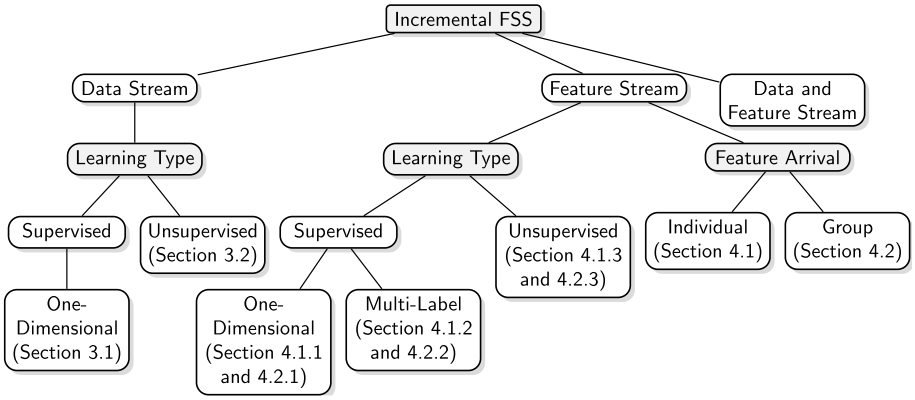


Fig. 4 Classification of the analyzed incremental FSS methods

group), will be studied. Additionally, some algorithms for contexts where both instances and features appear simultaneously are commented on.

As discussed in Sect. 1, the terminology used for FSS algorithms that work on dynamic data differs depending on the author. After a study of the literature, the following terms have been found to reference FSS algorithms for data streams:

- Dynamic feature selection (Barddal et al. 2017).
- Incremental attribute reduction (Hu et al. 2005).
- Incremental feature selection (Liang et al. 2014).
- Instance-based online streaming feature selection (Rahmaninia and Moradi 2018).
- Online feature selection (Wang et al. 2014).
- Standard online feature selection (Eskandari and Javidi 2016).

Proposals that can be applied to feature streams can be found as follows:

- Feature-based online streaming feature selection (Rahmaninia and Moradi 2018).
- Incremental attribute reduction (Jing et al. 2016).
- Incremental feature selection (Zeng et al. 2015).
- Online feature selection (Yu et al. 2014).
- Online streaming feature selection (Wu et al. 2013).
- Streaming feature selection (Zhou et al. 2005).
- Streamwise feature selection (Dhillon et al. 2010).

To encompass under a single term all the algorithms that perform FSS in an incremental manner, we think that it is appropriate to use *incremental feature subset selection*. In turn, the proposals could be additionally classified as *online feature subset selection* algorithms to be consistent with the analysis of online and incremental learning in Sect. 1.

3 Feature subset selection on data streams

FSS over data streams aims to determine which features to add to and delete from the current set of selected features with the arrival of new instances. In this section, several algorithms will be discussed for supervised learning tasks and for unsupervised environments where it is not possible to evaluate the feature relevance against a class variable.

3.1 Supervised learning

As mentioned in Fig. 1, a common approach to perform online FSS is to define weights for each feature that are updated with the arrival of new instances without having to repeatedly process or store instances already seen. These weights are then used to select the most important features. In this category, two algorithms that use regularization and truncation techniques are presented (Wang et al. 2014): *online feature selection by learning with full inputs* (OFS) and the *online feature selection by learning partial inputs* (OFS_p). Both fit a linear classifier and update it when a new instance is misclassified.

OFS assumes the full input of every new training instance to be received, including a binary class variable $Y \in \{-1, +1\}$. Then, it can update the model using two different approaches: (1) the *modified perceptron by truncation for OFS*, which, given a misclassified instance $(\mathbf{x}_t, y_t)$, updates the current weights $\mathbf{w}^t \in \mathbb{R}^d$ by simply increasing them with the vector $\mathbf{x}_t y_t$ (Perceptron rule), and (2) the *OFS via sparse projection*, which updates the weights by online gradient descent and projects them to an L_2 ball. After updating the weights, both methods truncate them to zero except for the b largest, where b is a predefined value. The first approach is a simple method that does not guarantee that discarded features have sufficiently small weights, which can result in many classification mistakes. The second proposal avoids this problem by projecting the weight vector to an L_2 ball before the truncation, so its norm is ensured to be bounded.

Meanwhile, OFS_p further limits the problem to cases where learners can access only a small and fixed feature set of the training instances, an interesting setting when the entire feature set for all instances is expensive to obtain. OFS_p does not simply choose b features with nonzero weights, but it defines the feature subset by alternating phases of exploration, where b features are chosen randomly, and exploitation, which chooses the b features whose weights are nonzero. In that way, the method avoids getting stuck in a configuration with poor classification performance.

Recent studies have focused on reducing the computational and memory costs of FSS algorithms for applications with large-scale and high-dimensional data. For example, Wu et al. (2017) presented a *second-order online feature selection* (SOFS) algorithm that seeks to improve previous first-order algorithms in terms of effectiveness, efficiency and scalability over high-dimensional sparse data. While first-order algorithms update the feature weights using only the first-order derivative information of the gradient, second-order proposals can also use second-order information, such as geometrical properties of data, to improve the feature selection. SOFS is based on the confidence-weighted method introduced by Dredze et al. (2008), which maintains not only feature weights but also an estimate of their confidence. The inclusion of this additional information may be positive for the performance of online FSS algorithms since it retains information about past instances. As a result, confidence can be used to guide the updating rule

of the feature weights, decreasing those of the less confident features. Different measures can be used to represent the confidence of the weights. In the case of SOFS, the weight vector is assumed to be modelled by a Gaussian distribution $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean vector $\boldsymbol{\mu}$ and a diagonal covariance matrix $\boldsymbol{\Sigma}$, parameters that are updated with the misclassification of new instances. The j-th entry in the diagonal of $\boldsymbol{\Sigma}$ represents the confidence in w_j. Therefore, the lower the value of $\boldsymbol{\Sigma}_{jj}$ is, the higher the confidence in the weight of feature j. Note that instead of drawing a weight vector $\mathbf{w}$ every time an instance is received, SOFS simply uses the mean vector $\boldsymbol{\mu}$ to perform predictions.

Exploiting the second-order information of feature weights implies an additional high computational cost. Thus, the authors reduced the complexity of SOFS by two means. First, only features with nonzero values in the new instance are updated and evaluated if a misclassification occurs. In this way, SOFS is linearly dependent on the number of nonzero features instead of on d, as OFS is. However, this advantage is limited to those problems where the data are sparse. Second, SOFS uses a max heap-based approach, i.e., it maintains a binary tree of features where the parent nodes have a covariance that is greater than or equal to that of their children. Selecting the most relevant features by searching in the covariance diagonal can be a time-consuming task. Therefore, this tree data structure allows more efficient retrieval of the b currently most confident weights without having to sort all of them in every step. The max heap combined with the monotonic decreasing property of the covariance, i.e., its value can only decrease or stay the same when it is updated, helps to further reduce the computational cost of the algorithm. Thus, when the weight and confidence of a certain feature in the tree are updated, the tree is modified only from the position of this feature towards its children since its parent is guaranteed to have a larger covariance. Additionally, those features that were not updated will not be part of the max heap since the covariance of the root node cannot increase. Then, the last step is to check whether the covariance of those features that were updated and are not in the tree is less than that of the root node. In such a case, the feature of the root node is removed and its weight is set to zero, while the tree is updated with the inclusion of this new feature with a lower covariance.

SOFS (like OFS) focuses on binary classification problems. Thus, Wu et al. (2017) extended it to handle a multi-class variable, meaning that $Y \in \{0, \ldots, k-1\}$ is a classification problem with k classes. This extension, which is referred to as SMOFS for simplicity, is based on using the one-vs-rest strategy. SMOFS essentially turns the multi-class problem into several binary problems, so $\mathbf{w} \in \mathbb{R}^{kd}$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{kd \times kd}$, and at a time t, the total confidence of the j-th feature is defined by a combination of each of its confidences in the different binary tasks.

FSS can be performed across multiple related tasks, which is known as *multi-task feature selection* (MTFS). The objective of multi-task learning is to simultaneously build models for tasks that are sufficiently different but still have commonalities, with the intention of obtaining more precise models than if they were obtained separately. For example, in spam filtering, users could need different features to label an email as spam; however, they would still share common features (Weinberger et al. 2009). A single classifier would perform correctly generalizing among different users but would fail to learn the specific preferences of a single user (Dredze and Crammer 2008). Then, learning all tasks simultaneously could improve the performance of each of their models, thanks to sharing their related information. Note that this learning problem differs from transfer learning, which aims to solve a single target task by transferring knowledge from a very similar source task. Transfer learning does not optimize the performance of

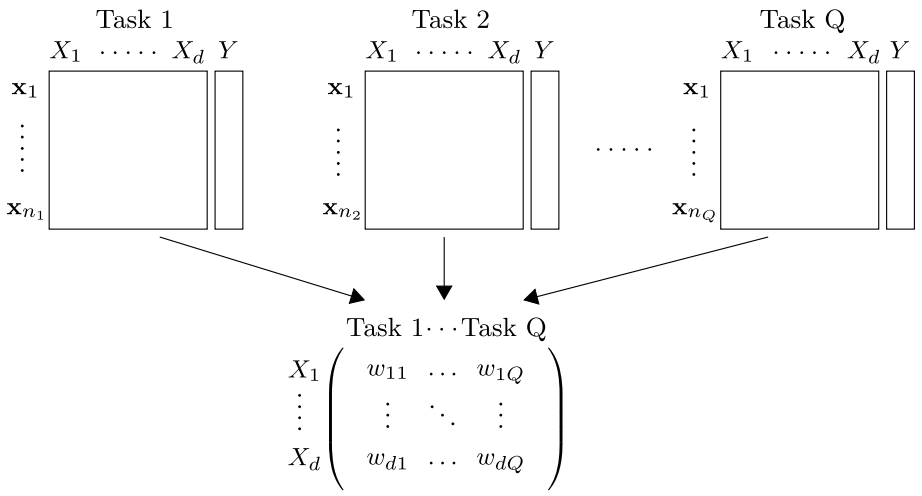


Fig. 5 Multi-task feature selection learning scheme

all the tasks involved and, therefore, does not allow the transfer of information among them in all possible directions (Torrey and Shavlik 2010).

The multi-task problem consists of Q tasks whose data come from the same space, with each task having n_q instances, $q = 1, \ldots, Q$. The entire dataset can be represented as $\mathcal{D} = \bigcup_{q=1}^{Q} \mathcal{D}_q$, where $\mathcal{D}_q = \{(\mathbf{x}_i^q, y_i^q)\}_{i=1}^{n_q}$ is a sample from distribution $\mathcal{P}_q$. The final task is to learn q functions, f_q, one for each task, such that $f_q(\mathbf{x}^q)$ approximates y^q as accurately as possible. These functions are parametrized by a weight vector, i.e., $f_q(\mathbf{x}^q) = (\mathbf{w}_q)^\top \mathbf{x}^q$, so most MTFS methods seek to learn a weight matrix $\mathbf{W}$, whose dimension is $d \times Q$ (see Fig. 5), by minimizing an empirical risk and a regularizer (Yang et al. 2013). This matrix defines the importance of each feature for the different tasks.

MTFS has been studied mainly from a batch perspective. However, online FSS can be applied in multi-task problems by using proposals such as the *dual-averaging for multi-task feature selection* (DA-MTFS) framework (Yang et al. 2013), which selects relevant features across tasks. At iteration t, DA-MTFS needs to receive Q instances, one for each task, with the objective of updating the weight matrix $\mathbf{W}^t$. This is performed by computing the subgradient $\mathbf{G}^t$ of a chosen loss function on the current weights, which is used to store an average of all the computed subgradients $\overline{\mathbf{G}}^t$ calculated until time t. The choice of the loss function depends on the problem to solve. For example, Yang et al. (2013) proposed the logit and hinge losses for binary classification and the square loss for regression tasks. Finally, the average subgradient is used along with a regularization term to compute the new weight matrix $\mathbf{W}^{t+1}$. DA-MTFS uses a mixed $L_{p,1}$-norm in the regularization term, i.e., a hybridization of the L_1-norm and an L_p-norm, to perform a joint regularization across multiple tasks. In other words, the mixed norm groups the weights of a specific feature for every task using an L_p-norm and applies an L_1-norm over the resultant vector. The general form of the mixed $L_{p,r}$-norm is computed over a matrix $\mathbf{W}$ as:

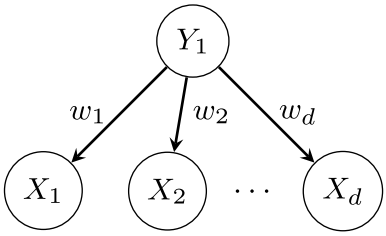


Fig. 6 Weighted naive Bayes structure

$$||\mathbf{W}||_{p,r} = \left(\sum_{i=1}^{d} ||\mathbf{w}_{i\bullet}||_p^r \right)^{\frac{1}{r}},$$

where $\mathbf{w}_{i\bullet}$ is the i-th row of $\mathbf{W}$. Three mixed norms were studied: (1) $p = 1$, which proportions sparse solutions but only observes the instances of each task individually, (2) $p = 2$, which uses the information across tasks simultaneously, with the inconvenience of resulting in non-sparse solutions and (3) a novel $L_{1/2,1}$-norm regularization, i.e., a linear combination of the L_1-norm and L_2-norm, which allows sparse solutions while still considering the information provided by all tasks.

DA-MTFS assumes the arrival of one instance per task in each iteration, which may not be realistic. The authors proposed not updating those tasks that do not receive an instance in a certain iteration; however, this would make them have less influence on the weights. Currently, there is no proposal that avoids this inconvenience. Additionally, it would be interesting to extend the online FSS algorithms to other multi-task settings, for example, to cases where there is no common set of features across all tasks due to outlier tasks (Gong et al. 2012). Finally, we would like to clarify that this online learning scenario differs from the lifelong learning problem, as the latter receives new tasks over time (no instances), which it tries to solve using knowledge acquired from previous tasks (Chen and Liu 2018).

Feature weights can also be included in naive Bayes classifiers to perform embedded FSS on data streams. Naive Bayes is an interesting probabilistic classifier algorithm based on applying Bayes' theorem with a strong conditional independence assumption between the features given the class variable. Despite its simplicity, the low complexity of naive Bayes makes it attractive for predicting data streams (Klawonn and Angelov 2006; Salperwyck et al. 2015). Naive Bayes can be improved by avoiding its attribute conditional independence assumption, which is not commonly held in real-world problems. This is possible by, for example, weighting the predictive variables. These algorithms are known as *weighted naive Bayes classifiers*, and they introduce a weight for each variable (see Fig. 6) to relax the attribute conditional independence assumption.

An online weighted naive Bayes proposal was introduced by Salperwyck et al. (2015), which we will refer to as OWNB. Although it does not focus on the online FSS problem by assigning feature weights, as previous algorithms do, the relevance of each variable is established. This algorithm defines a weight per class and per variable, computing them using stochastic gradient descent for a certain cost function. Thus, given a new training instance $(\mathbf{x}_i^t, y_i^t)$, the weights are updated as follows:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta \frac{\partial Cost(\mathbf{x}_i^t, y_i^t)}{\partial \mathbf{w}},$$

where $Cost(\cdot, \cdot)$ is a cost function and η is the learning rate.

Instead of relying on a single FSS algorithm, we could combine the knowledge extracted from different methods to define a more robust feature subset. This is the approach followed by the *multi-objective automated negotiation based online feature selection* (MOANOFS) system introduced by BenSaid and Alimi (2021), which identifies relevant and redundant features based on the feature weights assigned by different online FSS algorithms. This system is divided into two decision levels. The first level selects the most confident learners by means of a trust model (Das and Islam 2011). Subsequently, in the second level, relevant features are defined with a novel multilateral automated negotiation method according to the feature weights and prediction error reported by each of the learning algorithms selected in the first level. The final weight assigned to a feature will be determined by certain rules depending on the number of algorithms that find the feature relevant and the weight they report. This system can therefore be considered an ensemble technique, as it combines the outcomes of multiple learning algorithms to try to obtain a better result. Unfortunately, it should be noted that it is limited to FSS methods that are embedded in binary classifiers.

In a multi-class context, we can resort to the simpler *online bagging feature selectors* (OFS-Bag) and *online boosting feature selectors* (OFS-Boo) ensemble algorithms presented by Ditzler et al. (2017). As the names of the algorithms suggest, the ensembles are built using variants of online bagging and boosting (Oza and Russell 2001). These proposals do not rely on a negotiation method to define the final feature weights, but rather these are defined as a linear combination of the weights given by the base online FSS algorithms. Therefore, it is straightforward to apply techniques such as the one-vs-rest strategy (as the previous SMOFS does) in order to use binary classifiers in a multi-class classification problem. Although the authors report the best classification results when using the boosting approach in their experiments, it is worth noting that the possibility of parallelizing the learning of the base models when using the bagging-based algorithm may be of great interest in certain online learning environments.

Previous proposals do not explicitly define the relevance of features but rather adapt the currently selected features by monitoring the performance of the models and updating some feature weights. However, incremental FSS does not only involve the use of feature weighting approaches since other algorithms explicitly quantify the importance of the features for the class variable. This is the case of tree-based learning algorithms (Domingos and Hulten 2000; Bifet and Gavaldà 2009; Hulten et al. 2001; Gama et al. 2006), which decide the splitting feature based on, for example, mutual information or the Gini index. A well-known algorithm of this kind is the *very fast decision tree* (VFDT) (Domingos and Hulten 2000), which defines the best feature to perform a split by using the most recent instances that were received from the stream. VFDT may omit the use of some features in the decision tree, so it implements embedded FSS. The set of instances studied to select the splitting feature is given by the Hoeffding bound, from which the statistics for the splitting are extracted for each leaf node. In this way, instances are not stored and, therefore, they are not revisited.

The drawback of the VFDT algorithm is that it cannot efficiently deal with concept drift since the built tree is immutable and it is only possible to continue growing the tree in its leaves. Thus, the features selected for splits are never revoked. For those cases, the *concept-adapting very fast decision tree* (CVFDT) can be considered (Hulten et al. 2001).

CVFDT keeps the tree updated by replacing sub-trees that were outdated after a concept drift occurs. To do so, a sliding window of instances is considered. The oldest instances can be forgotten from the tree, while the newest instances are used to build alternate sub-trees and decide if they should replace the old ones.

The above algorithms have in common that they embed FSS into a model. However, we may be interested in a filter approach, which can be applied regardless of the model. A well-known algorithm of this kind is the *fast correlation-based filter* (FCBF), which defines the relevance and redundancy of features by analyzing their correlation (Yu and Liu 2003). If the correlation between a feature and a class variable is higher than a predefined threshold, then that feature is considered to be relevant. However, if its correlation with another feature is higher than that with the class variable, then it is removed since it is redundant. This is measured using the *symmetrical uncertainty*, which is a normalized version of the *mutual information* that does not favour features with more values. Given two variables X_1 and X_2, the symmetrical uncertainty between them is defined as:

$$SU(X_1, X_2) = 2\left(\frac{I(X_1, X_2)}{H(X_1) + H(X_2)}\right),$$

where the mutual information is:

$$I(X_1, X_2) = H(X_1) - H(X_1|X_2), \tag{1}$$

which measures the reduction in the entropy ($H(\cdot)$) of X_1 when X_2 is known.

FCBF is not an incremental algorithm since it considers that all training instances are available at the beginning when computing the relevance of each feature. However, the use of time windows allows us to apply the method to data streams. Articles such as Nguyen et al. (2012) prove the interest in applying this window version of FCBF on data streams, where it was embedded into a heterogeneous ensemble algorithm that is incrementally updated. Thanks to the inclusion of FCBF, the processing time to build the base models is reduced, while the accuracy of the ensemble is improved. This simple approach to adapt the traditional FSS algorithm to incremental scenarios is commonly used, and other famous algorithms, such as Relief (Kira and Rendell 1992) or its extensions (Kononenko 1994), could benefit from it.

3.1.1 Rough set-based FSS

One inconvenience of working with data streams is that the knowledge is incomplete since new instances appear over time. As explained in Sect. 2, rough set theory is ideal for this setting, as it needs only the dataset as input, and the objective of the attribute reduction is to generate minimal reducts that are sufficient to characterize the knowledge of a dataset. However, this process is an NP-hard task (Skowron and Rauszer 1992), so heuristic methods have been proposed. Most of the proposed algorithms follow the same approach, which consists of analyzing whether the current feature subset meets a certain criterion and, if not, the most significant features are included until the criterion is met. Afterwards, variables that do not compromise the criterion when removed are defined as redundant. We would like to emphasize that it is common to use the term *significance* when working with rough sets to refer to the importance/relevance of a feature. This section studies incremental FSS algorithms based on rough set theory, classifying them according to the heuristic used to compute the reducts.

Classical rough set theory can be directly applied to multi-label problems; however, this approach has several limitations (Li et al. 2016). No specific proposals for incremental attribute reduction for multi-label data streams were found, although articles such as Liu et al. (2018a) show that such an adaptation should not be complex. That said, any of the following algorithms are considered to be appropriate for multi-label tasks. In addition, due to the nature of this theory, practically all the proposed algorithms need to store the previously seen instances. This inconvenience can be avoided with the use of an appropriate time window, although it may not be trivial to define (Yang et al. 2018).

FSS based on the positive region One way to define a new reduct due to variation in the number of instances is to study the changes in the positive region. In this category, we find algorithms such as the *incremental attribute reduction algorithm based on the element set*, which we refer to as IARES (Hu et al. 2005). To understand IARES, first, the concepts of positive $\mathcal{P}$ and negative $\mathcal{N}$ elementary sets must be introduced. $\mathcal{P}$ contains all the condition classes of $\mathcal{U}/IND(\mathcal{X})$ that are consistent, i.e., all instances in these equivalence classes share the same value for the class variable. The opposite is true for the $\mathcal{N}$ set, in which all the condition classes of $\mathcal{U}/IND(\mathcal{X})$ are inconsistent, i.e., each of them has at least two instances with different values for the class variable. Knowing this, a subset $\mathcal{B} \subseteq \mathcal{X}$ is a reduct iff there are no collision equivalence classes on $\mathcal{B}$ in $\mathcal{P}$. This is an equivalence class that (1) shares the same values for the features in $\mathcal{B}$ with all condition classes in $\mathcal{P}$ but not the same class variable value or (2) shares the same values for features in $\mathcal{B}$ with one equivalence class in $\mathcal{N}$. The reason for avoiding collision equivalence classes on $\mathcal{B}$ in $\mathcal{P}$ is that, in other cases, $POS^{\mathcal{U}}_{\mathcal{B}}(Y) = POS^{\mathcal{U}}_{\mathcal{X}}(Y)$ would not hold, which would imply a loss of information since the positive region given by $\mathcal{X}$ is not preserved by $\mathcal{B}$.

When processing a new instance $\mathbf{z}_i$, IARES considers three scenarios: (1) there exists an equivalence class $\mathcal{E}_j \in \mathcal{N}$ to which $\mathbf{z}_i$ belongs, (2) there exists an equivalence class $\mathcal{E}_j \in \mathcal{P}$ to which $\mathbf{z}_i$ belongs, and (3) there is no equivalence class $\mathcal{E}_j$ to which $\mathbf{z}_i$ belongs. It is only in the third situation where the current reduct $\mathcal{S}^t$ may need to be updated since a new equivalence class $\mathcal{E}_{new}$ is created for $\mathbf{z}_i$ (and added to $\mathcal{P}$ since it is a unit set), which could be a collision equivalence class on $\mathcal{S}^t$ in $\mathcal{P}$. If this is the situation, IARES solves the collision by adding a feature (or features) from $\mathcal{X} \backslash \mathcal{S}^t$ that eliminates the collision, i.e., which makes $\mathcal{E}_{new}$ have different feature values with respect to any equivalence class on $\tilde{\mathcal{S}}^t = \mathcal{S}^t \cup X_i$ in $\mathcal{P}$. In this way, a transitional subset $\tilde{\mathcal{S}}^t$ is obtained that satisfies $POS^{\mathcal{U} \cup \{\mathbf{z}_i\}}_{\tilde{\mathcal{S}}^t}(Y) = POS^{\mathcal{U} \cup \{\mathbf{z}_i\}}_{\mathcal{X}}(Y)$ after adding $\mathbf{z}_i$ to the dataset. To avoid confusion, the notation $\tilde{\mathcal{S}}^t$ will be used to refer to the current state of the reduct, i.e., $\mathcal{S}^t$ modified by the inclusion or removal of features, which is prior to the final result $\mathcal{S}^{t+1}$. Finally, redundant features are removed from $\tilde{\mathcal{S}}^t$. A feature $X_i \in \tilde{\mathcal{S}}^t$ is considered to be redundant if $POS^{\mathcal{U} \cup \{\mathbf{z}_i\}}_{\tilde{\mathcal{S}}^t \backslash \{X_i\}}(Y) = POS^{\mathcal{U} \cup \{\mathbf{z}_i\}}_{\mathcal{X}}(Y)$. Once all redundant features are removed, the remaining features in $\tilde{\mathcal{S}}^t$ form the reduct $\mathcal{S}^{t+1}$.

Shu and Qian (2015) go a step further and propose the *incremental attribute reduction algorithm for the immigration of multiple objects* (IARM-I) for environments where multiple instances are added simultaneously to incomplete datasets, i.e., datasets where some feature values are missing for certain instances. IARM-I is able to process new instances that arrive by groups, something not possible with IARES, which receives groups but processes instances one by one. The interest in processing groups instead of single instances is based on the speed at which large volumes of data can be generated in some problems. Therefore, processing instances individually may be inefficient. When a group of instances $\mathcal{U}_a$ arrives, IARM-I compares the positive regions obtained from $\mathcal{U} \cup \mathcal{U}_a$ given the reduct $\mathcal{S}^t$ and the complete feature set $\mathcal{X}$. If they

do not have the same positive region, features from $\mathcal{X} \setminus \mathcal{S}^t$ are included in $\mathcal{S}^t$ until they do. The addition of features is performed in descending order of significance value in the current $\tilde{\mathcal{S}}^t$.

There is no common way to define the significance of a variable; however, it is usually divided into outer and inner significance, so the effect of adding (outer) or removing (inner) a feature from a set is assessed. Given a feature $X_i \notin \tilde{\mathcal{S}}^t$ and a measure $M(\cdot,\cdot)$, the outer significance of X_i in $\tilde{\mathcal{S}}^t$ relative to Y can be defined as:

$$sig^{outer}(X_i, \tilde{\mathcal{S}}^t, Y) = |M(\tilde{\mathcal{S}}^t, Y) - M(\tilde{\mathcal{S}}^t \cup \{X_i\}, Y)|. \tag{2}$$

If $X_i \in \tilde{\mathcal{S}}^t$, the inner significance can be obtained as:

$$sig^{inner}(X_i, \tilde{\mathcal{S}}^t, Y) = |M(\tilde{\mathcal{S}}^t, Y) - M(\tilde{\mathcal{S}}^t \setminus \{X_i\}, Y)|. \tag{3}$$

It is common to use outer significance to define the most important features to include in the reduct, while inner significance indicates the redundant features that should be removed. However, note that these approaches just compute the difference in a measure when a feature is considered or not. Therefore, this is not a practice common to all proposals.

In the case of Shu and Qian (2015), the variation in the cardinality of the positive region is used as the measure to compute the significance of the variables, i.e., $M(\tilde{\mathcal{S}}^t, Y) = card(POS^{\mathcal{U}}_{\tilde{\mathcal{S}}}(Y))$. Then, features with higher outer significance are included in the reduct until $POS^{\mathcal{U} \cup \mathcal{U}_a}_{\tilde{\mathcal{S}}^t}(Y) = POS^{\mathcal{U} \cup \mathcal{U}_a}_{\mathcal{X}}(Y)$. Once this condition is met, redundant features are removed from $\tilde{\mathcal{S}}^t$ if their inner significance in $\tilde{\mathcal{S}}^t$ is zero. The resultant $\tilde{\mathcal{S}}^t$ is the new reduct $\mathcal{S}^{t+1}$.

Previous algorithms expect new instances to be added to the dataset. However, FSS can also be performed when a set of instances $\mathcal{U}_r$ is removed from $\mathcal{U}$. The existence of these algorithms is justified by possible erroneous instances that need to be removed, for example, erroneous diagnoses of patients (Shu et al. 2019). The work done by Shu and Qian (2015) contemplates this possibility by proposing the *incremental attribute reduction algorithm for the emigration of multiple objects* (IARM-E), which is a reduced version of IARM-I in which only the existence of irrelevant features in the reduct is checked. The elimination of instances is also taken into account by Shu et al. (2019), where two novel algorithms are presented, one for the inclusion of new instances and another for the deletion of instances. These algorithms, which are known as the *incremental approach for feature selection in the decision system under multiple objects being added* (IFSA) and *under multiple objects being deleted* (IFSD), perform an incremental updating of the reduct using the dependency function. Each time a new set of objects is received or deleted, IFSA and IFSD incrementally update the dependency functions $\gamma^{\mathcal{U}'}(\mathcal{S}^t, Y)$ and $\gamma^{\mathcal{U}'}(\mathcal{X}, Y)$, which represent to what degree Y depends on $\mathcal{S}^t$ and $\mathcal{X}$, respectively, where $\mathcal{U}'$ is the set of instances after the addition or elimination. The degree of dependency compares the number of instances in the positive region with the number in the studied universe:

$$\gamma^{\mathcal{U}'}(\mathcal{S}^t, Y) = \frac{card(POS^{\mathcal{U}'}_{\mathcal{S}^t}(Y))}{card(\mathcal{U}')}. \tag{4}$$

The closer its value is to one, the more instances in the universe can be correctly classified with the available information. If $\gamma^{\mathcal{U}'}(\mathcal{S}^t, Y) \neq \gamma^{\mathcal{U}'}(\mathcal{X}, Y)$, the reduct is incremented with the sequential addition of features with the highest outer significance. Once

$\gamma^{\mathcal{U}'}(\mathcal{S}^t, Y) = \gamma^{\mathcal{U}'}(\mathcal{X}, Y)$, redundancies are removed. A feature $X_i \in \tilde{\mathcal{S}}^t$ is redundant if its inner significance in $\tilde{\mathcal{S}}^t$ is equal to zero. The outer and inner significances are computed as in Eqs. (2) and (3), where $M(\tilde{\mathcal{S}}^t, Y) = \gamma^{\mathcal{U}'}(\tilde{\mathcal{S}}^t, Y)$.

FSS based on the discernibility matrix The *incremental updating algorithm of attribute reduction set* (IUAARS) (Guan 2009) uses the discernibility matrix element set to report all the possible reducts at a certain moment and is able to process new instances that arrive by groups. The discernibility matrix indicates the subset of features that distinguish each pair of instances, and the discernibility matrix element set is formed by the nonempty elements of this matrix. Thus, when a new instance arrives, IUAARS updates the current reduct by analyzing the changes in this set due to inconsistencies or repetitions or because the instance was not seen before. If a change is necessary, IUAARS uses the set to find indispensable and dispensable features.

The *dynamic attribute reduction algorithm based on 0-1 integer programming* (Xu et al. 2011), which will be called DARIP for convenience, uses a similar approach. DARIP is based on creating a set of inequality constraints over a new instance set, where each constraint indicates the necessary features to discern each instance pair, therefore reporting the most relevant features. Although the authors considered DARIP to be an algorithm to address multiple instances simultaneously, it generates constraints by pairs of new instances, which in the end, implies analyzing each new instance one by one against the rest. The constraints grow quadratically with the number of instances; thus, preprocessing is introduced to remove redundant constraints. Once this is done, features in the remaining constraints form a reduct for the incoming instance set, which is used to increase $\mathcal{S}^t$ and obtain the final reduct $\mathcal{S}^{t+1}$. The disadvantage of DARIP is that it does not eliminate features from $\mathcal{S}^t$, so it cannot efficiently handle concept drift since features that become irrelevant or redundant are not removed. However, unlike previous rough set-based approaches, DARIP does not need to store previously seen instances.

FSS based on knowledge granularity Few algorithms use knowledge granularity to quantify the discernibility of features. Given that $\mathcal{U}/IND(\mathcal{X}) = \{\mathcal{E}_1, \ldots, \mathcal{E}_v\}$, the knowledge granularity of $\mathcal{X}$ is computed as:

$$GK^{\mathcal{U}}(\mathcal{X}) = \sum_{i=1}^{v} \frac{card(\mathcal{E}_i)^2}{card(\mathcal{U})^2}. \tag{5}$$

As can be deduced from Eq. (5), the smaller the knowledge granularity is, the stronger the discernibility since the partitioning is finer and, therefore, the size of the equivalence classes is smaller. This measure can be used to dynamically find a reduct when adding or deleting instances since the goal is to use a feature subset that maintains the knowledge granularity achieved with the complete feature set.

Notable members of this category include the *updating attribute reduction algorithm when adding* (UARAO) and *deleting* (UARDO) *some objects with a multi-granulation view* (Jing et al. 2017), two algorithms that focus on improving the efficiency when addressing large-scale datasets that are increased or reduced dynamically, respectively. To do so, they adapt the knowledge granularity formula to be incrementally updated for datasets with a multi-granulation view, which is based on dividing a dataset into m small sub-tables (small granularities), i.e., $\mathcal{U} = \cup_{i=1}^{m} \mathcal{U}_i$. Approaches to determine the size of the sub-tables were studied by Liang et al. (2012). The equivalence classes from each sub-table are obtained and merged to obtain the equivalence classes from the complete dataset. This strategy enables the equivalence classes to be obtained in a more efficient

way since computing them directly from a large dataset is more costly in terms of computational time and space.

UARAO follows the same approach as the previously studied algorithms. Upon the arrival of a new instance set $\mathcal{U}_a$, the algorithm checks whether the conditional knowledge granularity of Y under $\mathcal{X}$ and $\mathcal{S}^t$ in $\mathcal{U}_a$ is the same, i.e., $GK^{\mathcal{U}_a}(Y|\mathcal{S}^t) = GK^{\mathcal{U}_a}(Y|\mathcal{X})$. The conditional knowledge granularity is computed as:

$$GK^{\mathcal{U}_a}(Y|\mathcal{S}^t) = GK^{\mathcal{U}_a}(\mathcal{S}^t) - GK^{\mathcal{U}_a}(\mathcal{S}^t \cap Y). \tag{6}$$

If the knowledge granularity is the same, the reduct is not altered since the equivalence classes obtained in $\mathcal{U}_a$ by $\mathcal{S}^t$ and $\mathcal{X}$ are identical and, therefore, the discernibility is the same in $\cup_{i=1}^{m}\mathcal{U}_i \cup \mathcal{U}_a$. If the equality is compromised, relevant features from $\mathcal{X} \setminus \tilde{\mathcal{S}}^t$ are included in the reduct based on their outer significance measure in $\tilde{\mathcal{S}}^t$, which is computed with Eq. (2), where $M(\tilde{\mathcal{S}}^t, Y) = GK^{\cup_{i=1}^{m}\mathcal{U}_i}(Y|\tilde{\mathcal{S}}^t)$. When $GK^{\mathcal{U}_a}(Y|\mathcal{S}^t) = GK^{\mathcal{U}_a}(Y|\mathcal{X})$, the final reduct $\mathcal{S}^{t+1}$ is obtained after the removal of redundant features, i.e., any feature in $\mathcal{S}^{t+1}$ that would not alter the knowledge granularity.

In the case of removing a set of instances from $\mathcal{U}$, UARDO simply evaluates whether the features in $\mathcal{S}^t$ are still relevant and non-redundant and removes those that do not alter the knowledge granularity when discarded.

FSS based on entropy information Entropy is a common uncertainty measure that can be used in rough set theory to define the significance of features. Some common entropies that can be applied within rough set theory are Shannon's (Wierman 1999), complementary (Liang et al. 2002) and combination (Qian and Liang 2008). Let $\mathcal{B} \in \mathcal{X}$ and $\mathcal{U}/IND(\mathcal{B}) = \{\mathcal{E}_1, \ldots, \mathcal{E}_v\}$; then, the Shannon $H^{\mathcal{U}}(\mathcal{B})$, complementary $E^{\mathcal{U}}(\mathcal{B})$ and combination $CE^{\mathcal{U}}(\mathcal{B})$ entropies of $\mathcal{B}$ for the instance set $\mathcal{U}$ can be defined, respectively, as:

$$H^{\mathcal{U}}(\mathcal{B}) = -\sum_{i=1}^{v} \frac{card(\mathcal{E}_i)}{card(\mathcal{U})} \log \frac{card(\mathcal{E}_i)}{card(\mathcal{U})},$$

$$E^{\mathcal{U}}(\mathcal{B}) = \sum_{i=1}^{v} \frac{card(\mathcal{E}_i)}{card(\mathcal{U})} \left(1 - \frac{card(\mathcal{E}_i)}{card(\mathcal{U})}\right),$$

and

$$CE^{\mathcal{U}}(\mathcal{B}) = \sum_{i=1}^{v} \frac{card(\mathcal{E}_i)}{card(\mathcal{U})} \left(1 - \frac{C^2_{\mathcal{E}_i}}{C^2_{\mathcal{U}}}\right),$$

where $C^2_{\mathcal{E}_i} = card(\mathcal{E}_i)(card(\mathcal{E}_i) - 1)/2$ is the number of indistinguishable instance pairs in $\mathcal{E}_i$.

The conditional version of these three entropies $ME(\cdot|\cdot)$ was provided by Liang et al. (2014) for data streams, so they are updated incrementally with changes in the equivalence classes due to the inclusion of new instances. Furthermore, they introduced the *group incremental algorithm for reduct computation* (GIARC), an algorithm that uses these measures to find a reduct when a new instance group is received.

Upon the arrival of a group of instances $\mathcal{U}_a$, GIARC updates the current reduct only if instances in $\mathcal{U}_a$ are indistinguishable using the features in $\mathcal{S}^t$ but distinguishable when using $\mathcal{X}$. This occurs when the conditional entropies $ME(\cdot|\cdot)$ (Shannon, combination or complementary) of $\mathcal{S}^t$ and $\mathcal{X}$ relative to Y are not the same, i.e., $ME^{\mathcal{U}_a}(Y|\mathcal{S}^t) \neq ME^{\mathcal{U}_a}(Y|\mathcal{X})$. In that case, the features from $\mathcal{X} \setminus \mathcal{S}^t$ with the highest outer significance in $\mathcal{S}^t$ are included

in the reduct until $ME^{\mathcal{U}\cup\mathcal{U}_a}(Y|\tilde{\mathcal{S}}^t) = ME^{\mathcal{U}\cup\mathcal{U}_a}(Y|\mathcal{X})$, where $ME^{\mathcal{U}\cup\mathcal{U}_a}(\cdot|\cdot)$ represents one of the three novel formulas used to recompute the entropy after adding $\mathcal{U}_a$. The outer significance is computed as in Eq. (2), defining $M(\tilde{\mathcal{S}}^t, Y)$ as the previously selected incremental conditional entropy. Once relevant features are added, redundant ones are removed by checking whether their inner significance in $\tilde{\mathcal{S}}^t$ is zero. This is computed with Eq. (3), employing the same conditional entropy as in the outer significance. The resulting set is the new reduct $\mathcal{S}^{t+1}$.

Experiments in Liang et al. (2014) proved that GIARC produces results similar to those of algorithms based on the positive region, such as IARES, but the computational time is always shorter.

3.1.2 FSS for rough set extensions

Rough set theory was extended with proposals that attempted to overcome its shortcomings. One of the main problems is that it is not convenient to handle hybrid (discrete and continuous) attributes, performing efficiently only if the attributes are discrete. This problem can be avoided by discretizing continuous features, although this approach can cause information loss (Zeng et al. 2015). Thus, it may be more interesting to use fuzzy rough sets, a fuzzy generalization of rough sets that allows handling uncertainty when data are real-valued (Dubois and Prade 1990). Fuzzy rough set theory introduces the fuzzy similarity relation, which measures how similar two instances are. Thus, instances are grouped into equivalence classes using soft boundaries depending on their similarities (Cornelis et al. 2008).

Two recent algorithms for fuzzy rough set-based FSS were presented by Yang et al. (2018). These proposals adapt the reduct in an incremental manner using the relative discernibility relations of each feature and the feature set, which are updated every time a new set of instances arrives. The first novel algorithm is the *first incremental version for fuzzy rough set-based feature selection* (IV-FS-FRS-1), which checks whether the discernibility relation of the original feature set and that obtained with the reduct is the same. When it is not, new features are added to the reduct until they are equal. Finally, features that do not alter the previous equality when removed are discarded. The second novel algorithm is the *second incremental version for fuzzy rough set-based feature selection* (IV-FS-FRS-2), which follows the same strategy, except that it begins finding a reduct when there are no more new instance sets. Clearly, IV-FS-FRS-2 has the advantage of being faster than IV-FS-FRS-1; however, it cannot provide a feature subset at any time.

Another extension of rough set theory is variable precision rough set theory, which is used to overcome the sensitivity to noise and errors in data. Defining the $\mathcal{B}$-lower approximations of $\mathcal{H}$ using instances that share the same feature and class variable values may be a strict requirement, which does not reflect possible errors, such as in the observation of the data. Therefore, variable precision rough set theory includes a degree of uncertainty that relaxes the requirements to admit instances in the $\mathcal{B}$-lower approximation, thereby allowing a level of misclassification. The smaller this degree is, the smaller the boundary region (Ziarko 1993).

The first incremental attribute reduction algorithms for variable precision rough sets are the *incremental algorithm for β-upper distribution reduct* (Incremental-U) and *for β-lower distribution reduct* (Incremental-L) (Chen et al. 2016). These algorithms use discernibility matrices to dynamically find a reduct when a new instance is received.

Table 3 Supervised learning algorithms for FSS on data streams

Name	Description	Class variable	Process groups	Instance removal	No instance revisit	References
OFS and OFS$_p$	Use instances with full and partial inputs to update feature weights	Binary	✗	✗	✓	Wang et al. (2014)
SOFS and SMOFS	Second-order algorithms that use a MaxHeap-based approach	Binary and Multi-class	✗	✗	✓	Wu et al. (2017)
DA-MTFS	Performs online FSS over multiple related tasks simultaneously	Depends[a]	✗	✗	✓	Yang et al. (2013)
OWNB	Constructs a weighted naive Bayes classifier	Multi-class	✗	✗	✓	Salperwyck et al. (2015)
MOANOFS	Online ensemble FSS algorithm with a multi-objective negotiation technique	Binary	✗	✗	✓	BenSaid and Alimi (2021)
OFS-Bag and OFS-Boo	Bagging- and boosting-based online FSS algorithm	Multi-class	✗	✗	✓	Ditzler et al. (2017)
VFDT	Decision tree algorithm for data streams with embedded FSS	Multi-class	✗	✗	✓	Domingos and Hulten (2000)
CVFDT	Extension of VFDT that is able to handle concept drifts	Multi-class	✗	✗[b]	✗	Hulten et al. (2001)
FCBF[c]	Filter-based algorithm that employs symmetrical uncertainty to define relevant and redundant features	Multi-class	✓	✗[b]	Depends[d]	Yu and Liu (2003)
Rough set-based						
IARES	Studies equivalence classes to determine the reducts	Multi-class	✗	✗	✗	Hu et al. (2005)
IARM-(I/E)	Uses positive region on incomplete decision systems	Multi-class	✓	✓	✗	Shu and Qian (2015)
IFS(A/D)	Compares positive regions to find reduct in incomplete decision systems	Multi-class	✓	✓	✗	Shu et al. (2019)
IUAARS	Updates reduct by using the discernibility matrix element set	Multi-class	✓	✗	✗	Guan (2009)
DARIP	Based on 0-1 programming to determine the reduct through inequality constraints	Multi-class	✗[e]	✗	✓[f]	Xu et al. (2011)
UAR(AO/DO)	Uses knowledge granularity to find a reduct with a multi-granulation view	Multi-class	✓	✓	✗	Jing et al. (2017)

Table 3 (continued)

Name	Description	Class variable	Process groups	Instance removal	No instance revisit	References
GIARC	Incrementally updates information entropies to define reducts	Multi-class	✓	✗	✗	Liang et al. (2014)
IV-FS-FRS-(1/2)	Fuzzy rough set-based algorithms based on the relative discernibility relations	Multi-class	✓	✗	✗	Yang et al. (2018)
Incremental-(U/L)	Attribute reduction for variable precision rough sets which use discernibility relation matrices	Multi-class	✗	✗	✗	Chen et al. (2016)

[a]Different loss functions can be employed depending on the learning task

[b]Does not explicitly allow the removal of instances. It forgets old ones with time windows

[c]Batch algorithm that can be implemented with time windows

[d]The time window could be formed by old and new instances or only by new ones

[e]Needs an instance group to generate constraints, but instances are analysed one by one

[f]New instances are analysed more than once, but they are not stored for future iterations

In conclusion, Table 3 summarizes the analyzed supervised FSS algorithms for data streams. For each algorithm, the following are indicated: (1) the type of class variables the algorithm was designed for, (2) whether the algorithm can process a new group of instances, (3) whether the algorithm allows the elimination of previously received examples and (4) whether the algorithm re-analyses past instances.

3.1.3 Distributed FSS

Previous proposals follow a centralized approach, i.e., the FSS is performed by just one process in a single machine. It could be interesting to address big data processing from a distributed perspective since these algorithms can meet the necessary performance requirements by using multiple computing units. For instance, Fong et al. (2016) attempted to overcome computational and memory problems with the use of nature-inspired metaheuristics, specifically swarm algorithms. However, this approach is based on using a batch accelerated particle swarm optimization and defining the feature subset by maximizing the fitness obtained with an incremental classifier. Thus, it may not be able to efficiently handle a real-time data stream. Particle swarm optimization is one nature-inspired metaheuristic that could be applied to incremental FSS (Diao et al. 2013), but there is a lack of other nature-inspired approaches to this problem, such as genetic (Leardi et al. 1992; Oh et al. 2004), harmony search (Diao and Shen 2010, 2012) and particle swarm optimization variants such as the competitive swarm optimizer (Gu et al. 2018).

It is worth noting the distributed FSS technique for multi-label classification problems presented by Gonzalez-Lopez et al. (2020), which is based on the estimation of mutual information measures. Although it does not propose an online algorithm, we believe it may be of interest when processing high-dimensional data streams with time windows. Its main idea is to compute the mutual information between variables in a distributed environment with Apache Spark, as this is a computationally expensive task in a multi-label classification problem, especially with high-dimensional data.

3.2 Unsupervised learning

Performing FSS over unsupervised data faces the challenge of not being able to quantify the relevance of features by using the information provided a class variable. Three main approaches can be used to define the feature relevance: (1) consider the capability of the features to preserve the structure of the original data, (2) define cluster indicators with a clustering algorithm and perform supervised FSS, and (3) select those features with the highest or lowest correlation between them (Solorio-Fernández et al. 2020). In this section, we discuss some algorithms that perform unsupervised FSS in a data streaming context.

One of the first unsupervised FSS proposals for data streams is StreamFeatWeight (Huang et al. 2015), also known as FSDS. The key idea of this algorithm, to enable the application to data streams, is to use matrix sketching to generate a low-rank approximation, defined as a matrix sketch, of all the observed data (or the data contained in a time window) up to time t. In other words, instances received by time t are reduced into a matrix of a predefined rank k that most faithfully represents the original data. This rank-k approximation is updated with the inclusion of new data, which enables the algorithm to adapt to concept drift and make only one pass over the data. FSDS performs this step with a modified version of the *frequent-directions* algorithm (Liberty 2013). Therefore, singular value

decomposition is performed on the cosine similarity matrix of the matrix sketch augmented with new data to define its spectrum, i.e., eigensystem, and the new approximation. Then, the leading k eigenvectors, i.e., those corresponding to the k largest eigenvalues, are used as the target in a regression analysis with regularization to report new feature coefficients. These coefficients are finally used to obtain the importance score of each feature. The intuition behind this approach is that the leading eigenvectors contain information about the structure of the instance distribution, so the features that better predict the eigenvectors are those that have a stronger capability to maintain this structure (Zhao et al. 2010).

FSDS can be classified as a *spectral feature selection* algorithm, which comprises methods that define the relevance of features by their ability to preserve the structure of the original data. The main idea is to analyze the consistency of the features with the graph obtained from the similarities between instances. Specifically, the consistency between the features and the spectrum of a matrix obtained from the similarity is measured (Zhao and Liu 2011). For example, the *SPEC* framework uses the eigensystem of its normalized Laplacian matrix (Zhao and Liu 2007). The similarity can be computed with different measures, which can include or not include information about a class variable.

FSDS assumes that all the data come from a single data stream. However, data could be generated from multiple sources with distinct feature sets. One example is dividing image information into two views, visual and metadata, a problem known as multi-view learning. The main difference of multi-view learning with respect to the multi-task problem is that, in the former, different views provide complementary information about the same instances, while in the latter, each task has its own instances that share a common feature space. Different datasets have co-occurring n instances in n_v views, $\mathcal{D}_v = \{\mathbf{x}_i\}_{i=1}^{n}, v = 1, 2, \ldots, n_v$, where $\mathcal{D}_v \in \mathbb{R}^{n \times d_v}$ represents the dataset of the v-th view and d_v represents its feature dimension. Views can have high dimensionality; thus, FSS is used to obtain a feature subset from each view based on the information provided by the different views to improve the selection. In addition, multi-view data often include a large number of instances, so this information could be processed in a streamwise manner to avoid memory problems. The *online unsupervised multi-view feature selection* (OMVFS) algorithm is the first approach that covers the unsupervised multi-view problem from an online perspective (Shao et al. 2016). OMVFS includes FSS into an online nonnegative matrix-based clustering algorithm, processing data chunks in an online manner and aggregating their information into different small matrices without storing all the previous data. To do so, a parameter is used to define the maximum size of the matrices and remove excess old information; i.e., a time window is used to reduce the memory requirements without losing the capability to capture concept drifts. The main objective of OMVFS is to generate a cluster indicator matrix that integrates information from all the views and a feature selection matrix for each of the n_v views, which is updated in each iteration with new and some old data. It is from this feature selection matrix that the importance of each feature in every view is obtained.

A summary of the unsupervised FSS algorithms for data streams is provided in Table 4.

4 Feature subset selection on feature streams

This section focuses on different approaches to perform FSS on feature streams, from algorithms that evaluate new features one by one in one-dimensional tasks to algorithms that evaluate feature groups in multi-label environments. Individual and group FSS for unsupervised environments is also studied. Although, at first glance, it may be thought that the

Table 4 Unsupervised learning algorithms for FSS on data streams

Name	Description	Process groups	Instance removal	No instance revisit	References
FSDS	Uses matrix sketching and regression analysis with regularization to weight features at every time step	✓	✗	✓	Huang et al. (2015)
OMVFS	Performs unsupervised FSS over streaming multi-view data	✓	✗	✗	Shao et al. (2016)

inclusion of new instances is more common, most recent studies have focused on working with feature streams.

The arrival of new features could imply that those features already selected are no longer the most relevant, i.e., weakly relevant, and even redundant. Therefore, it is necessary to adapt the subset of selected features over time to always contain the most relevant and non-redundant features. FSS algorithms for feature streams are useful not only in those scenarios where our set of features is, theoretically, infinite but also in those cases where the generation of features is expensive, so it could be more interesting to include them in our model progressively (Perkins and Theiler 2003). A well-known example is the detection of a large number of small craters on the surface of Mars from high-resolution images (Ding et al. 2011). This analysis involves tracking a large number of texture features that cannot be pre-generated in a reasonable time; thus, they may be considered as soon as they are obtained.

4.1 Individual FSS

In this section, we analyze supervised and unsupervised FSS algorithms that process new features at the individual feature level.

4.1.1 Supervised one-dimensional learning

We start by discussing scenarios where data are labelled, so it is possible to compute the relevance of the features with respect to one or several class variables. First, the algorithms that work with one-dimensional learning problems and feature streams are introduced.

The feature stream problem, introduced by Perkins and Theiler (2003), considers how to efficiently solve FSS problems in a dynamic environment with new features appearing one by one while training instances remain static. The authors proposed an adaptation of the *grafting* algorithm (Perkins et al. 2003) for feature streams in Perkins and Theiler (2003), which uses a fast gradient-based heuristic to quickly find which features are most likely to improve an existing model. This technique can be used with models parametrized by a weight vector that is subject to Lasso regularization, so the inclusion of new features will result in the augmentation of the model with new weights. Grafting is presented along with two different models, a linear and non-linear model, ensuring that it combines the speed of filters and the accuracy of wrappers.

Another interesting algorithm to consider is *alpha-investing* (Zhou et al. 2005), which is an (adaptive) complexity penalty method based on dynamically adapting a threshold α^t to control which new features are added in the future. Alpha-investing is implemented with a model that is able to take features sequentially and report a p value that represents the probability that a new feature is included in the model when it should be discarded, i.e., a false positive. The p value of a feature is associated with a t-statistic, whose value is equivalent to the difference in the log-likelihood of the data given a model with the feature and another without it. Then, a new variable is included in the model if its p value is less than α^t. This threshold represents the probability of including spurious features at time step t and is dynamically computed using the wealth of the current iteration w^t:

$$\alpha^t = \frac{w^t}{2t}.$$

The wealth indicates the currently acceptable number of false positives. Therefore, if a feature is accepted, the wealth is increased, while it is reduced if the feature is rejected. The alpha-investing algorithm is based on the *information-investing* algorithm (Ungar et al. 2005), a similar approach that adds a new feature if it reduces the entropy of the model sufficiently with respect to an adaptive threshold.

As with multi-view learning on data streams, our problem could involve not only one feature stream but multiple of them. In this case, we can use an extension of alpha-investing, the *multiple streamwise feature selection* (MSFS) algorithm (Dhillon et al. 2010). MSFS stores the wealth for each of the streams and selects the next feature from the stream with the most permissive threshold since it should be the most beneficial for the model.

Alpha-investing has the advantage over grafting in that it does not need prior information about the global feature set. Grafting requires this information to choose a good regularization parameter, a major limitation for some scenarios since we can be considering, theoretically, infinite streams of features (Wu et al. 2013). However, both algorithms suffer from the so-called nesting effect, which means that they cannot discard redundant features that were previously selected (Pudil et al. 1994) since they do not re-evaluate them. To avoid the nesting effect, we can use the *online streaming feature selection* (OSFS) algorithm (Wu et al. 2013), which is capable of selecting relevant features in real time and removing redundant ones in two major steps. First, an online relevance analysis is performed to determine whether a new feature is relevant for the class variable. Second, and the novelty of this approach, online redundancy analysis is performed to identify and remove previously selected features that became redundant due to the inclusion of a new relevant feature. The OSFS algorithm uses conditional independence and dependence tests implemented with the G^2 test to determine relevant and redundant features. Thus, the conditional dependency test is computed between the new feature and the class variable to determine if it is relevant. A feature is identified as redundant if there exists a subset from the set of selected features that makes this feature and the class variable conditionally independent. Thus, this process is performed for every selected feature, removing those that become redundant. OSFS uses the Markov blanket criterion to define redundant features, which guarantees that a removed feature will still be considered redundant even after the posterior removal of others. However, it is computationally expensive to check all possible subsets of features every time a new feature is received. Therefore, the authors assume a predefined size for the checked subsets. This would reduce the time complexity but will also affect the results. Depending on the characteristics of the problem being addressed, it may be necessary to compromise between obtaining more reliable results and the efficiency of the algorithm.

The maintenance of a good feature subset in a real-time environment demands efficient approaches, and in the case of OSFS, the analysis of redundancies can be time consuming. The redundancy analysis was divided into two parts in a novel algorithm called fast-OSFS (Wu et al. 2013) to accelerate the computation. The first part of the redundancy analysis only checks whether a new relevant feature X_t is redundant with respect to any other subset of selected features. In that case, the new feature is discarded, and the algorithm moves directly to the next feature of the stream. Otherwise, fast-OSFS checks whether the inclusion causes other features to become redundant, just as OSFS does. However, in this case, the computational cost of conditional independence tests is reduced since only those subsets from the selected features that include the new feature X_t are considered.

OSFS and fast-OSFS not only avoid the nesting effect but also do not need prior information about the feature set. However, they do have a disadvantage since the use of

conditional independence tests entails a need for a large number of instances to obtain reliable results. This is even more important in this environment since the set of features grows over time (Javidi and Eskandari 2019).

Despite significant improvements in the online FSS research area, Yu et al. (2014) noted that previous algorithms, such as alpha-investing and fast-OSFS, do not perform efficiently enough when applied to data of extremely high dimensionality. This is also the case of rough set-based and fuzzy rough set-based algorithms for feature streams commented on later in this section, such as CIE-OSFS or FRSA-IFS-HIS(AA). Therefore, proposals such as the *scalable and accurate online approach for feature selection* (SAOLA) focus on the scalability of FSS methods (Yu et al. 2014). SAOLA reduces the computational cost of identifying non-relevant and redundant features by employing only pairwise comparisons to compute the correlations between attributes, where the mutual information is used as a metric (see Eq. (1)).

SAOLA considers a new feature X_t as relevant for the class variable Y if their mutual information $I(X_t, Y)$ is greater than a predefined relevance threshold. In that case, SAOLA checks whether the feature introduces redundancy. Thus, if the mutual information between X_t and any other already selected feature $X_j \in \mathcal{S}^t$ is greater than or equal to $I(X_t, Y)$ or $I(X_j, Y)$, one of the features contains at least as much information about the class variable as the other. Therefore, the feature with lower relevance (mutual information) to the class variable is removed, i.e., the redundant feature. This approach was proposed by Yu and Liu (2004) as a way to approximate the Markov blankets that still guarantees that the revisiting of discarded features is not necessary.

SAOLA is more efficient than OSFS or fast-OSFS, even if the latter defines a maximum size for the subsets to evaluate. Experiments in Yu et al. (2014) indicate that SAOLA does not appear to significantly improve the prediction accuracy, while fast-OSFS reports smaller feature subsets. However, fast-OSFS could require days to run in datasets of extremely high dimensions, while SAOLA would need only a few seconds or minutes to find a solution. In addition, the use of mutual information has the advantage, with respect to the rough set approaches, of being able to handle discrete and continuous features.

A disadvantage of SAOLA is that it requires a predefined relevance threshold to determine relevant features, and it is not trivial to define an appropriate value. This problem is avoided with *online stream feature selection based on mutual information* (OSFSMI) and the *online stream feature selection based on mutual information with fixed number of features* (OSFSMI-*k*), two algorithms that also employ mutual information to assess the relevance and redundancy of features (Rahmaninia and Moradi 2018). Both algorithms start by discarding irrelevant new features, which are those whose mutual information with the class variable is zero. If a new feature is considered relevant, redundant features are detected in $\mathcal{S}^t$ by evaluating their effectiveness. The effectiveness of a feature X_i is computed by taking into account its relevance and redundancy with respect to the remaining features $\mathcal{S}^t \setminus X_i$:

$$\lambda(X_i, \mathcal{S}^t) = I(X_i, Y) - \beta \sum_{X_j \in \mathcal{S}^t \setminus X_i} \frac{I(X_j, Y)}{H(X_j)} I(X_i, X_j),$$

where the parameter $\beta \in (0, 1]$ controls the redundancy penalty. Features with lower effectiveness values than the newly included feature are removed, while the effectiveness values of features that still remain are updated. In the case of OSFSMI, the redundancy analysis stops when no more features of lower effectiveness are found or if only one feature

remains. OSFSMI-k returns the best feature subset of a predetermined size k. Then, the feature subset $\mathcal{S}^{t+1}$ is obtained, and both algorithms wait for the next feature.

OSFSMI and OSFSMI-k appear to give more importance to new features since they cannot be discarded upon arrival if their relevance is greater than zero. A major problem with this behaviour is that weak relevant features would be accepted and possibly discarded in the next iteration, resulting in a waste of resources. Thus, it could be more interesting to include a threshold, as SAOLA does, since expecting the mutual information to be zero is a very strict requirement. Experiments in Rahmaninia and Moradi (2018) show that these algorithms can find feature subsets that provide higher classification accuracy than previous proposals while obtaining similar or better run times. In addition, OSFSMI is more stable than those algorithms that also perform redundancy analysis, i.e., more similar subsets of features are obtained on independent executions and with different feature orders, and smaller feature subsets than those of algorithms such as SAOLA are produced.

Note that hybridizations of previously proposed algorithms can take advantage of their benefits. For example, a new hybrid approach between the *wind driven dynamic optimization algorithm* (WD2O) (Boulesnane and Meshoul 2017), a nature-inspired dynamic optimization algorithm, and the previously explained OSFS algorithm was introduced by Boulesnane and Meshoul (2018). This novel algorithm is the *dynamic online streaming feature selection* (DOSFS), and the authors demonstrate, through several experiments, that the use of dynamic optimization for the selection of a feature subset significantly improves the accuracy obtained by OSFS for specific problems while taking advantage of the latter's speed. This improvement is the result of the recovery of the features discarded by OSFS that is performed by WD2O. The final feature subset is a combination of the subsets found by OSFS and WD2O.

Rough set-based FSS As feature streams maintain a fixed number of instances, the application of rough set-based proposals can be of great interest. However, until recently, FSS was not considered from a rough set perspective, and the *dimension incremental algorithm for reduction computation* (DIA-RED) was among the first proposals (Wang et al. 2013). DIA-RED uses the Shannon, combination or complementary entropy to incrementally adapt the reduct without recomputing the entropies for the whole dataset each time a new set of features is received. DIA-RED includes incremental mechanisms that enable us to update the existing entropies based on the changes in the condition and decision classes. It follows a similar strategy to that of GIARC (see Sect. 3.1.1), which considers the use of the same entropies and outer and inner significance measures to determine relevant and redundant features.

DIA-RED considers only the information contained in the positive region. However, this approach could result in defining sets of variables as independent due to noise. Given a set of instances $\mathcal{U}$, the *online streaming noise-resistant-aided rough set attribute reduction using significance analysis* (OS-NRRSAR-SA) algorithm (Eskandari and Javidi 2016) defines the reduct using the degree of dependency between attributes $\gamma^{\mathcal{U}}(\cdot,\cdot)$ (see Eq. (4)) and a novel noise-resistant dependency measure that additionally evaluates the boundary region:

$$\rho^{\mathcal{U}}(\mathcal{S}^t, Y) = \frac{\tau^{\mathcal{U}}(\mathcal{S}^t, Y) + \gamma^{\mathcal{U}}(\mathcal{S}^t, Y)}{2}. \tag{7}$$

The measure $\tau^{\mathcal{U}}(\cdot,\cdot)$ includes information about the proximity of the boundary and positive region and the possibility of transferring instances from one to another by removing noisy instances.

If the degree of dependency between Y and the selected features $\mathcal{S}^t$ is less than one, i.e., the dataset is inconsistent using $\mathcal{S}^t$, OS-NRRSAR-SA includes a new feature X_t in $\mathcal{S}^t$ as long as (1) its outer significance is greater than zero or (2) the noise-resistant dependency of Y on X_t is nonzero. The outer and inner significances of features are defined in Eqs. (2) and (3), employing the degree of dependency as a measure and normalizing the difference in the case of the inner significance. If the degree of dependency is one, i.e., the dataset is consistent using $\mathcal{S}^t$, the new feature X_t is not simply ignored since it can replace a subset of $\mathcal{S}^t$ that becomes redundant due to its inclusion. This subset is defined based on the inner significance, and if its size is greater than one, it is replaced by X_t, making the reduct more compact while keeping the dataset consistent. In the case that the subset is a unit set, the feature with the highest noise-resistant dependency is retained. The number of subsets to evaluate grows exponentially with the size of $\mathcal{S}^t$ when looking for the largest redundant one. Therefore, sequential backward elimination is also proposed to obtain a more efficient solution. However, the removal order that is employed affects the result.

Experiments in Eskandari and Javidi (2016) compare the performance of OS-NRRSAR-SA, grafting, information-investing, fast-OSFS and DIA-RED, with OS-NRRSAR-SA showing superiority in terms of the compactness of the selected subsets, computational time and classification accuracy achieved with the selected feature subsets.

OS-NRRSAR-SA was later extended (Javidi and Eskandari 2019) because the original approach is not efficient enough when the number of selected features is not small. This is because it extracts equivalence classes considering all the currently selected features, which is a computationally expensive task. To make this approach slightly more scalable, the proposed extension seeks to obtain more compact results by including a new filter method that removes redundancies before the significance analysis. As this new step may be computationally expensive, a user-defined parameter is proposed to define the maximum size of the redundant subsets. The extension was compared against the original OS-NRRSAR-SA, and some improvements were detected in terms of compactness, run time and classification accuracy. Although the extension adds a new step to OS-NRRSAR-SA, the run time was reduced for certain datasets because of the more compact results.

As a last note about OS-NRRSAR-SA, the same strategy was used in another approach called SFS-RS (Javidi and Eskandari 2018), whose only difference is that it was conceived with six different measures of dependency, including the noise-resistant measure of Eq. (7). This work was useful to determine the effects of using different measures.

Other approaches use the knowledge granularity to incrementally update the reduct. This is the case of Jing et al. (2016), where matrix-based and non-matrix-based algorithms called the *matrix-based incremental reduction algorithm* (MIRA) and *incremental algorithm for reduct computation* (IARC), respectively, are proposed. This research seeks not only to analyze the benefits of incremental algorithms with respect to batch versions but also to prove the inefficiency of matrix-based algorithms for large datasets. MIRA and IARC use the same strategy: first, given a new group of features $\mathcal{G}^t$, they check that the new granularity $GK^{\mathcal{U}}(Y|\mathcal{X} \cup \mathcal{G}^t)$ (see Eqs. (5) and (6)) is still the same as the granularity of the current reduct $GK^{\mathcal{U}}(Y|\mathcal{S}^t)$. If this is not the case, the features with the highest outer significance from $\{\mathcal{X} \cup \mathcal{G}^t \setminus \mathcal{S}^t\}$ are included in the reduct until the granularities are equalized. The final step is to remove features from the reduct as long as their removal does not alter the previous equality, i.e., redundant features, to obtain the new reduct $\mathcal{S}^{t+1}$.

The only difference between MIRA and IARC is the method used to calculate the knowledge granularity. MIRA makes use of relation matrices, which may not be efficient enough for large datasets due to the memory and computational time needed. A more efficient method is presented in IARC, which computes the knowledge granularity with

a non-matrix-based approach using the new partitions that occur in the dataset with the inclusion of new features.

As features are unknown until they are received, the order of their arrival may affect the obtained feature subsets. The *online streaming feature selection algorithm based on conditional information entropy* (CIE-OSFS) considers this problem and is robust to changes in the order because of the implementation of a sorting mechanism (Wang et al. 2017). CIE-OSFS divides its strategy into two phases. First, an independence test checks whether a new feature is relevant to the class variable. In the second phase, redundancies are removed by searching for a reduct in the set of relevant features. During the redundancy analysis, features are sorted by their correlation coefficient against the class variable, so CIE-OSFS is more stable to changes in the feature arrival order. The new reduct $\mathcal{S}^{t+1}$ is generated by computing the outer significance of each relevant feature, which is calculated using Eq. (2), where $M(\cdot,\cdot)$ is the Shannon conditional entropy. The features are considered not redundant only if their outer significance in the current reduct is greater than zero.

FSS for rough set extensions As noted in Sect. 3.1.2, some extensions of classical rough set theory were proposed to overcome its problems. This includes its inefficiency in handling continuous features, which can be solved using fuzzy rough sets. In the case of feature streams, Zeng et al. (2015) proposed two fuzzy rough set approaches for incremental FSS on hybrid information systems (HIS), i.e., datasets that include different types of attributes, such as binary, continuous, categorical and set-valued, a common scenario in real-world problems. These novel algorithms are known as the *fuzzy rough sets approach for incremental feature selection in HIS under one attribute being added* (FRSA-IFS-HIS(AA)) and *under one attribute being deleted* (FRSA-IFS-HIS(AD)). Both algorithms use a novel hybrid distance function $HD(\cdot,\cdot)$ to compute the distance between instances with hybrid and incomplete variables. This distance is based on the Euclidean distance, but it uses five different equations to compute the value difference of each type of variable. Given two instances $\mathbf{x}_i$ and $\mathbf{x}_j$, their distance is computed as follows:

$$HD(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{r=1}^{d} vd(x_{ir}, x_{jr})^2}, \tag{8}$$

where $vd(\cdot,\cdot)$ computes the value difference depending on the value type of x_{ir} and x_{jr}.

FRSA-IFS-HIS(AA) is based on the positive region, and it uses the fuzzy dependency to define relevant and redundant features. This algorithm takes into account the feature order in the reduct, so those with a higher degree of dependency (with the class variable) are located in the first positions. This is done to modify only those elements of $\mathcal{S}^t$ that are in a lower position than the new feature X_t since the lower the dependency value is, the lower the importance. When X_t is received, FRSA-IFS-HIS(AA) first incrementally computes its fuzzy relations, which are obtained by using the Gaussian kernel function with the hybrid distance of Eq. (8). These relations are then used to obtain fuzzy positive regions necessary to compute the dependencies between Y and the feature subsets (see Eq. (4)). If X_t increases the dependency for a certain subset of $\mathcal{S}^t$, it will be included in $\mathcal{S}^{t+1}$, and the remaining features can be added from its position in the reduct. These features are added in descending order of dependency value, as long as they increase the dependency of the current reduct by more than a certain threshold; otherwise, they are considered redundant.

FRSA-IFS-HIS(AD) updates the reduct after the elimination of a feature $X_{del} \in \mathcal{S}^t$. Then, following the same approach as FRSA-IFS-HIS(AA), it merely removes X_{del} and

adds the remaining features from its position in the reduct to obtain $\mathcal{S}^{t+1}$. Previously, only unnecessary features were removed by the algorithms. However, this work introduces the possibility of removing features due to an external decision, even if they were considered to be relevant. We think this possibility may be useful, for example, in the case that a feature was detected as inappropriately collected. Thus, the feature can be eliminated and the reduct incrementally updated. Notably, FRSA-IFS-HIS(AD) (as well as FRSA-IFS-HIS(AA)) stores all the variables that were available at some point, which for memory reasons is not good practice when working with the incremental addition of new features. Therefore, how to remove more efficiently relevant features while recovering those that were redundant due to the features being removed remains an open issue.

The neighborhood rough set is another extension of the classical rough set theory commonly used to handle continuous features without requiring a discretization (Hu et al. 2008). This is achieved by using a neighborhood relation to define the lower and upper approximations, rather than equivalence classes that group instances with the same values for the studied attributes. Traditionally, this neighborhood relation is based on the use of a distance function to find for each instance the k nearest neighbor instances (k-nearest neighborhood relation) or all those within a given δ distance (δ neighborhood relation) (Zhou et al. 2019b). The K-OFSD (Zhou et al. 2017) is an example of a neighborhood rough set-based algorithm, which uses the k-nearest neighborhood relation to perform an online FSS that improves the separability between majority and minority classes of imbalanced datasets.

The main drawback of K-OFSD is that it requires the hyperparameter k to be tuned beforehand. An inconvenience that also occurs with the δ hyperparameter of the δ neighborhood relation. Therefore, with the intention of avoiding this problem, Zhou et al. (2019b) introduced a gap neighborhood relation, which automatically selects the number of neighbors considered based on the distribution of the surrounding instances. Basically, it places the neighbors of an instance in ascending order of distance and defines a cutoff when there is a certain "gap" (distance value) between two consecutive instances. This neighborhood relation was devised to be used in the novel OFS-A3M algorithm, which performs online FSS by analyzing the degree of dependency of a new feature on the class variable (see Eq. 4) and the effect on the dependency of the currently selected feature subset when this new feature is included.

Subsequently, the above authors proposed the OFS-Density algorithm (Zhou et al. 2019a) with the main objective of including a more aggressive removal of redundant features than OFS-A3M. Instead of only performing a redundancy analysis when a new relevant feature does not improve the feature subset dependence degree (a rare occurrence in a real scenario), OFS-Density considers this task when the improvement is smaller than a given threshold. This can certainly increase the number of features detected as redundant, but it also implies setting a new hyperparameter. In addition, OFS-Density introduces a density neighborhood relation that automatically selects the number of neighbors. Similar to the gap neighborhood relation, this new relation sorts the neighbors in ascending order of distance and sets a cutoff not only taking into account the distances, but also the number of instances already selected as neighbors. According to the experiments conducted by Zhou et al. (2019a), OFS-Density selects feature subsets that are significantly smaller than those of OFS-A3M, and classifiers trained with those subsets report, in general, higher accuracy. However, this comes at the cost of significantly slower performance, a handicap in certain online scenarios.

4.1.2 Supervised multi-label learning

Most state-of-the-art algorithms were designed for one-dimensional supervised learning problems. However, multi-dimensional learning has multiple applications in real-world problems, such as document analysis (Zhu et al. 2005), bioinformatics (Elisseeff and Weston 2001) and music (Trohidis et al. 2011). Any information can be classified into several class variables simultaneously. Thus, it is necessary to consider online FSS algorithms for this environment. A simple solution would be to use a transformation approach in which independent one-dimensional algorithms are used for each class variable. However, this approach would ignore the possible relations between class variables, information that could be important to find the best subset of features. Multi-label learning over feature streams brings several new challenges to overcome, such as label correlation, high dimensionality, class imbalance and label-specific features (Lin et al. 2017). Note that the multi-label problem is a subtype of the multi-dimensional problem, where all the class variables are binary.

To the best of our knowledge, Lin et al. (2017) proposed the first online FSS solution for multi-label problems, the *multi-label streaming feature selection* algorithm, which we refer to as ML-SFS. This algorithm focuses on the label correlation and high-dimensionality problems in an online scenario and introduces the fuzzy mutual information $FI(\cdot,\cdot)$ as the evaluation criterion to define relevant and redundant features in a multi-label context. Correlations between class variables and features are stored in two separate similarity matrices, which include all influences of one variable on the others. The fuzzy mutual information is computed using these similarity matrices, so the mutual dependence between a feature X_t and the label space $\mathcal{Y}$ is defined as follows:

$$FI(X_t, \mathcal{Y}) = -\frac{1}{n} \sum_{i=1}^{n} \log \frac{card([\mathbf{x}_i]_{X_t}) \cdot card([\mathbf{x}_i]_{\mathcal{Y}})}{n \cdot card([\mathbf{x}_i]_{X_t} \cap [\mathbf{x}_i]_{\mathcal{Y}})},$$

where $[\mathbf{x}_i]_{X_t}$ is the fuzzy equivalence class of $\mathbf{x}_i$ under X_t, which is determined with the similarity degrees between $\mathbf{x}_i$ and every instance under feature space X_t. Therefore, given a predefined relevance threshold δ, a new feature X_t is classified as relevant if $FI(X_t, \mathcal{Y}) > \delta$. If this is the case, ML-SFS removes possible redundancies caused by the inclusion of X_t in $\mathcal{S}^t$, considering a feature $X_i \in \mathcal{S}^t$ as redundant if its fuzzy mutual information is lower, i.e., $FI(X_t, \mathcal{Y}) > FI(X_i, \mathcal{Y})$.

A different approach to address multi-label problems is given in the *multi-label online streaming feature selection algorithm based on spectral granulation and mutual information* (ML-OSMI) (Wang et al. 2018), which seeks to capture the correlations among labels by transforming the label set $\mathcal{Y} = \{Y_1, \ldots, Y_l\}$ into a new set of multi-class variables $\mathcal{Y}' = \{Y'_1, \ldots, Y'_h\}$ with lower dimensionality, i.e., $h < l$. This step is performed only once before the relevance and redundancy analysis since the label set is assumed to be fixed. Labels are first clustered using spectral clustering with cosine similarity. Thus, each cluster is formed by labels with a high correlation. Then, each cluster is transformed into a multi-class variable by applying the label power set (LP) framework. After the label transformation, when ML-OSMI receives a new feature X_t, its relevance against each of the created class variables $Y'_i \in \mathcal{Y}'$ is assessed using the normalized mutual information for continuous variables:

$$nI(X_t, Y'_i) = \frac{2 \cdot \int \int p(X_t, Y'_i) \log \frac{p(X_t, Y'_i)}{p(X_t)p(Y'_i)} dX_t dY'_i}{-\int p(X_t) \log p(X_t) dX_t - \int p(Y'_i) \log p(Y'_i) dY'_i}.$$

If the normalized mutual information of X_t and any class variable Y'_i is greater than a predefined threshold, then X_t is considered relevant and included in $\mathcal{S}^t$. In such a case, its inclusion can produce redundancy. A feature $X_i \in \mathcal{S}^t$ is redundant if its significance on any class variable Y'_j given another feature X_j is zero. This significance is defined as the maximum conditional mutual information between X_i and each class variable in $\mathcal{Y}'$ given X_j.

The LP framework is useful to reduce the problem dimensionality or even to apply one-dimensional algorithms to multi-dimensional datasets. However, LP-based methods can suffer from class-imbalance problems if the number of labels is large (Wang et al. 2018) and they imply working with multi-class variables with high cardinality.

Rough set-based FSS Rough set-based FSS algorithms designed specifically to address multi-label feature streams are also available. This is the case for the *online multi-label streaming feature selection based on neighborhood rough set* (OM-NRS) algorithm (Liu et al. 2018a), which has the advantage of supporting continuous features since it is based on using the neighborhood rough set theory to define positive regions. OM-NRS starts by measuring the outer significance of a new feature X_t in $\mathcal{S}^t$ relative to the label set $\mathcal{Y}$ to determine its relevance. To do so, Eq. (2) is used with a multi-label adaptation of the degree of dependency (see Eq. (4)), which is defined as:

$$\gamma^{\mathcal{U}}(\mathcal{S}^t, \mathcal{Y}) = \frac{\sum_{k=1}^{l} card(POS^{\mathcal{U}}_{\mathcal{S}^t}(Y_k))}{l \cdot card(\mathcal{U})}.$$

X_t is included in $\mathcal{S}^{t+1}$ if its outer significance is greater than zero. Otherwise, online redundancy analysis is performed since there exists at least one feature in $\mathcal{S}^t$ that is redundant with X_t. In this case, and to reduce the number of calculations, it is first checked whether $\gamma^{\mathcal{U}}(X_t, \mathcal{Y})$ is less than a predefined relevance threshold, rejecting X_t in such cases since it is considered weakly relevant. If this does not hold, OM-NRS has to evaluate X_t against every feature $X_i \in \mathcal{S}^t$, discarding the feature with a lower degree of dependency with $\mathcal{Y}$ as long as $sig^{outer}(X_t, \mathcal{S}^t, \mathcal{Y}) = 0$. This last step continues until all X_i are evaluated or X_t is discarded.

Table 5 summarizes the discussed algorithms.

4.1.3 Unsupervised learning

FSS algorithms can also be applied over feature streams with no classes. A typical example is social media data, where features are usually generated dynamically and it is expensive to collect label information (Li et al. 2015).

The *unsupervised streaming feature selection* (USFS) framework is one of the first FSS algorithms for feature streams to handle unsupervised data (Li et al. 2015). USFS performs FSS through a regression model using link information in social media. Therefore, it works on datasets with n linked data instances, whose link information is encoded into a matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ such that if $M_{ij} = 1$, instances $\mathbf{x}_i$ and $\mathbf{x}_j$ are linked. USFS extracts a predefined number s of social latent factors from $\mathbf{M}$ to which each instance is associated with a certain probability. This information is included for each instance in a vector $\boldsymbol{\pi}_i \in \mathbb{R}^s$, so $\boldsymbol{\Pi} = [\boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_n]$. For example, a social latent factor could be hobbies or job positions shared between people. These factors should have a relationship with certain features; for

Table 5 Supervised learning algorithms for individual FSS on feature streams

Name	Description	Class variable	References
Grafting	Uses fast gradient-based heuristic to find the most likely feature to improve a model	Depends[a]	Perkins and Theiler (2003)
Alpha-investing	Uses the p value of new features to determine their inclusion in a model	Depends[a]	Zhou et al. (2005)
Information-investing	Adds new features to a model if they reduce the entropy sufficiently with respect to a threshold	Depends[a]	Ungar et al. (2005)
MSFS	Extension of alpha-investing that works simultaneously with multiple feature streams	Depends[a]	Dhillon et al. (2010)
(fast-)OSFS	Identifies relevant and redundant features by using conditional independence and dependence tests	Multi-class	Wu et al. (2013)
SAOLA	Reduces the computational cost of identifying redundant features by computing correlations via pairwise comparisons with mutual information	Multi-class	Yu et al. (2014)
OSFSMI(-*k*)	Uses mutual information to discard new features and compute their effectiveness	Multi-class	Rahmaninia and Moradi (2018)
DOSFS	Hybridization of the WD2O dynamic optimization algorithm and OSFS	Multi-class	Boulesnane and Meshoul (2018)
ML-SFS	Performs online relevance and redundancy analysis considering the correlations between labels	Multi-label	Lin et al. (2017)
ML-OSMI	Captures correlations among labels by transforming them into a set with lower dimensionality	Multi-label	Wang et al. (2018)
Rough set-based			
DIA-RED	Incremental attribute reduction algorithm proposed to be used with three different entropy measures	Multi-class	Wang et al. (2013)
OS-NRRSAR-SA and extensions	Uses the degree of dependency with different dependency measures	Multi-class	Eskandari and Javidi (2016); Javidi and Eskandari (2019, 2018)
MIRA and IARC	Matrix and non-matrix based attribute reduction algorithms that make use of knowledge granularity	Multi-class	Jing et al. (2016)
CIE-OSFS	Defines reduct using conditional information entropy and independence test. It is robust to changes in feature arrival	Multi-class	Wang et al. (2017)
FRSA-IFS-HIS(AA)/(AD)	Fuzzy rough set-based algorithms for hybrid datasets. FRSA-IFS-HIS(AD) considers the removal of features from the dataset	Multi-class	Zeng et al. (2015)
K-OFSD	Neighborhood rough set-based algorithm using the *k*-nearest neighborhood relation	Multi-class	Zhou et al. (2017)
OFS-A3M	Neighborhood rough set-based algorithm using a new gap neighborhood relation	Multi-class	Zhou et al. (2019b)
OFS-Density	Neighborhood rough set-based algorithm using a new density neighborhood relation	Multi-class	Zhou et al. (2019a)

Table 5 (continued)

Name	Description	Class variable	References
OM-NRS	Based on using neighborhood rough set theory to compute the positive region and a multi-label version of the degree of dependency	Multi-label	Liu et al. (2018a)

[a]Different models can be used with the algorithm

Table 6 Unsupervised learning algorithms for individual FSS on feature streams

Name	Description	References
USFS	Selects the most informative features for social media feature streams by making use of the link information of the data	Li et al. (2015)
UFSSF	Representative features of each cluster, generated with an extension of the k-means algorithm, are selected as the most informative features	Almusallam et al. (2018)

example, age could be related to hobbies, so they are used to decide if a new feature X_t should be included in the model. In some sense, the missing information about class variables is replaced with these social latent factors, so relevant features are defined by their ability to differentiate them. Then, FSS is performed through several regression models with regularization, where the social latent factors are the dependent variables, i.e., one regression model for each social latent factor, so the obtained regression coefficients at time step t, $\mathbf{W}^t \in \mathbb{R}^{t\times k}$, represent the importance of every feature on each social latent factor. Thus, USFS performs FSS by solving at time t the minimization problems for each social latent factor $\boldsymbol{\pi}_{\bullet i}$ (i-th column of $\boldsymbol{\Pi}$):

$$\begin{aligned} min_{\mathbf{w}^t_{\bullet i}} \mathcal{J}(\mathbf{w}^t_{\bullet i}) = \frac{1}{2}||\mathbf{X}^t\mathbf{w}^t_{\bullet i} - \boldsymbol{\pi}_{\bullet i}||^2_2 + \alpha||\mathbf{w}^t_{\bullet i}||_1 \\ + \frac{\beta}{2}||\mathbf{w}^t_{\bullet i}||^2_2 + \frac{\gamma}{2}||(\mathbf{X}^t\mathbf{w}^t_{\bullet i})^T(\mathbf{L}^t)^{\frac{1}{2}}||^2_2, \end{aligned} \tag{9}$$

where $\mathbf{X}^t \in \mathbb{R}^{n\times t}$ is the available data, $\mathbf{L}^t$ is a Laplacian matrix extracted from a similarity graph of the instances, and α, β and γ are parameters to control the penalty terms of Lasso regularization (second term) and elastic net regularization (third term) and to balance the link information and feature information (fourth term), respectively.

To define a new feature X_t as relevant or not, USFS first computes for each social latent factor $\boldsymbol{\pi}_{\bullet i}$ the derivative of Eq. (9) with respect to $\mathbf{w}^{t+1}_{\bullet i}$, which is an augmentation of $\mathbf{w}^t_{\bullet i}$ with a nonzero value for X_t. If the derivative of $\boldsymbol{\pi}_{\bullet i}$ is greater than the parameter α, the inclusion of X_t helps to reduce the objective function in Eq. (9) when predicting $\boldsymbol{\pi}_{\bullet i}$. Thus, X_t is included in the model, and Eq. (9) is optimized with the current weights using the Broyden-Fletcher-Goldfarb-Shanno algorithm. This approach may cause some regression coefficients to shrink to zero; therefore, a feature is not included in S^{t+1} iff $\mathbf{w}^{t+1}_{\bullet i}$ is an empty vector.

A disadvantage of USFS is that it does not consider the update of the link information $\mathbf{M}$. The authors justify this strategy by noting that this information does not change as often as new features are received. However, this may not be the case for some applications.

The USFS framework has two other limitations: it is valid only for those problems where link information can be established and the dataset grows with the inclusion of new features, i.e., features are not totally discarded but their weights are set to zero, which could lead to memory problems. The *unsupervised feature selection for streaming features* (UFSSF) algorithm avoids these issues by extending the k-means algorithm to include linearly dependent similarity measures and to work with streaming features (Almusallam et al. 2018). UFSSF clusters incoming features and selects a representative feature for each cluster. Therefore, the subset of features in one cluster is approximated by only one feature, and the original set of features is reduced to equal the number of clusters. A new feature X_t is assigned to the cluster

with which it has the highest similarity to the centroid, and it is selected as the representative feature of that cluster at time $t + 1$ if its similarity to the centroid (updated with the information of X_t) is higher than that of the representative feature at time t. UFSSF was designed to be used with three similarity measures, the Pearson correlation coefficient, the least squares regression error and the maximal information compression index, which have the advantage of not being sensitive to the feature order and the scatter of feature distributions.

UFSSF incrementally computes the centroids as a weighted mean of the features that were assigned to their clusters, giving more importance to those features that were received more recently. This algorithm is therefore indicated for problems where the most recent features are more relevant. Other inconveniences are that the number of clusters is expected to be provided and, therefore, the size of the subset of selected features must be predefined, and that the solution is highly dependent on the initial clusters. Note that UFSSF could be an efficient solution in a streaming scenario, but it simply reports a feature subset using linear dependent measures.

Table 6 summarizes all the unsupervised individual FSS algorithms for feature streams that were analyzed in this section.

4.2 Group FSS

Features may exhibit group structures; however, previous algorithms evaluate them individually. In this section, proposals that exploit this information for supervised and unsupervised learning are reviewed. Group FSS chooses feature groups that are relevant, in the case of supervised learning, to class variables, removing irrelevant features and avoiding redundancies both at the individual and group feature levels. There are several real-world problems where it is preferable to evaluate features in a group manner. For example, in image analysis, there are certain properties of images, such as colour information, that are described by a set of different features (Wang et al. 2015). Therefore, group FSS should not be confused with the simultaneous addition of new features that is performed by algorithms such as DIA-RED or MIRA (see Sect. 4.1.1). These latter proposals do not consider the information provided by the group structures in which the features are received.

4.2.1 Supervised one-dimensional learning

The *online group feature selection* (OGFS) algorithm may be the first approach that takes into account the group structure to perform online FSS (Wang et al. 2015). This method is based on performing two phases, the intra-group and the inter-group selection. In the intra-group selection, each feature of an incoming group $\mathcal{G}^t$ is processed individually to select a subset $\tilde{\mathcal{G}}^t$ by using spectral feature selection. Traditional spectral feature selection was adapted for the online environment, introducing a criterion that has to be satisfied to select a feature from an incoming group. This criterion seeks to include new features that increment the between-class while reducing the within-class distances between instances. This first phase evaluates the features individually. Thus, the inter-group selection phase defines an optimal subset $\hat{\mathcal{G}}^t \subseteq \tilde{\mathcal{G}}^t$ that reduces the redundancies between features of different groups. This step is performed by using linear regression with Lasso regularization, which results in setting to zero the coefficients of features considered redundant.

Another algorithm to consider is *group feature selection with streaming features* (GFSSF) (Li et al. 2013), which, in this case, uses mutual information to select and discard features or groups. When a new group of features arrives, a feature-level selection

Table 7 Supervised learning algorithms for group FSS on feature streams

Name	Description	Class variable	References
OGFS	Uses spectral feature selection to perform intra-group selection, while Lasso regression selects a final subset among different groups	Multi-class	Wang et al. (2015)
GFSSF	Performs FSS at individual and group feature levels by using mutual and conditional mutual information	Multi-class	Li et al. (2013)
group-SAOLA	Adaptation of SAOLA that is able to evaluate incoming groups of features	Multi-class	Yu et al. (2016b)
OMGFS	Allows online group FSS over multi-label feature streams	Multi-label	Liu et al. (2018b)

step uses the mutual information between the features in this group to select a subset of relevant ones, while redundancies are discarded. Once this subset is defined, a group-level selection phase is performed to evaluate the subset against already selected groups. This step determines whether the information provided by the subset is more than the penalty and if previously included groups should be removed. This penalty is defined based on the number of features, so large groups that provide less information for the class variable are more likely to be discarded. Unlike OGFS, GFSSF replaces or removes feature groups completely, i.e., once a group is included, its features are only discarded if the complete group is eliminated. As GFSSF uses conditional mutual information to define redundancies, this approach is useful for reducing the number of comparisons to be made. However, this could lead to a loss of important features contained in a deleted group.

Finally, there are also adaptations of algorithms that evaluate features individually so that they can handle incoming feature groups. This is the case of group-SAOLA (Yu et al. 2016b), which extends the SAOLA algorithm. Group-SAOLA checks the relevance of a new feature group by analyzing the mutual information between each of its features and the class variable. If the mutual information for all the new features is less than or equal to a predefined threshold, the group of features is discarded. Otherwise, the redundancies between features within the group and between this group and previous ones are removed. Group-SAOLA eliminates existing redundancies between features of different groups, so the groups are not eliminated unless all their features are redundant. Nevertheless, this algorithm uses a more scalable solution than GFSSF since it performs pairwise comparisons of the features instead of computing conditional mutual information. Regarding the OGFS algorithm, the experiments performed in Yu et al. (2016b) achieve similar or, in most cases, higher prediction accuracies using the solutions proposed by group-SAOLA, which always produces smaller feature subsets. Additionally, group-SAOLA is more efficient than OGFS on datasets of extremely high dimensionality.

4.2.2 Supervised multi-label learning

The aforementioned group FSS algorithms were created for one-dimensional problems. In the multi-label setting, we can use algorithms such as the *online multi-label group feature selection* (OMGFS) (Liu et al. 2018b). OMGFS starts by performing an online group selection when a new feature group $\mathcal{G}^t$ is received, where $\mathcal{G}^t$ is classified as relevant or not by comparing its correlation with the label set $\mathcal{Y}$ and a predefined threshold δ. This correlation

is computed using a normalized multi-label neighborhood mutual information, which takes into account the neighborhood of each instance induced by the variables under comparison. If the correlation of $\mathcal{G}^t$ with $\mathcal{Y}$ is larger than δ, then $\mathcal{G}^t$ is added to a buffer pool. This buffer pool is increased with new strongly and weakly relevant groups until reaching its predefined maximum capacity. At that moment, an inter-group selection detects the most relevant features and redundancies by defining interaction weights between features from the different selected groups. Each interaction weight represents the relationship of two features with respect to the class variable in such a way that the smaller the weight is, the more information the features provide about the class variables separately than together. The feature with the highest relevance, i.e., multi-label neighborhood mutual information, with $\mathcal{Y}$ is used to compute the interaction weight with the rest of the features. This feature and those with interaction weights larger than zero are included in the final subset.

Table 7 summarizes the supervised group FSS algorithms on feature streams.

4.2.3 Unsupervised learning

We did not find any interesting method that performs group feature selection in an unsupervised feature stream context. Nevertheless, it could be possible to adapt the strategies followed by algorithms such as FSDS (see Sect. 3.2), OGFS or Group-SAOLA (see Sect. 4.2.1) to the unsupervised scenario. For example, spectral feature selection with an unsupervised measure and regularized regression could be used in an intra-group phase to select the most relevant features from an incoming group. Then, the feature redundancy among groups could be reduced by means of information theory measures. This idea would require further research to determine for which real-world problems it could be useful.

5 Feature selection on data and feature streams

Several approaches that allow incremental FSS, either for data streams or feature streams, have been discussed. However, the appearance of new instances and features can occur simultaneously in real-world problems. An example is text clustering, where the number of documents (instances) and vocabulary (features) could evolve over time (Zhang et al. 2015). This is, of course, a more complex situation where the algorithm should adapt the feature selection depending not only on the available features at a certain moment but also on the evolution of their values given newly received instances.

To the best of our knowledge, no attention was given to this topic until the proposal of the *sparse trapezoidal streaming data learning* (STSD) algorithm (Zhang et al. 2015). STSD dynamically maintains a linear model for binary classification, whose weights are updated based on whether a misclassification is produced, and includes an embedded FSS by projection and truncation. This algorithm is based on the strategy followed by the OFS algorithm (see Sect. 3.1), but in this case, the feature dimension of an incoming training instance is larger than or equal to that of the current classifier. Given a new instance $\mathbf{z}_t = (\mathbf{x}_t, y_t) \in \mathbb{R}^{d^t}$, where $y_t \in \{-1, +1\}$ and d^t is the size of the feature space at time t, and a weight vector $\mathbf{w}^t \in \mathbb{R}^{d^{t-1}}$, STSD predicts the label of $\mathbf{x}_t$ using only the values $\tilde{\mathbf{x}}_t$ of the features that were considered in the current classifier, i.e., those features that have nonzero weight in $\mathbf{w}^t$. Therefore, $\mathbf{x}_t = (\tilde{\mathbf{x}}_t, \hat{\mathbf{x}}_t)$, where $\hat{\mathbf{x}}_t$ is the values of features with zero weights. This approach will result in a loss l^t, computed with the hinge loss function, which will be used to update the weights according to the following rule:

Table 8 Supervised learning algorithms for FSS on data and feature streams

Name	Description	Class variable	No instance revisit	References
STSD	Maintains a linear classifier, which is updated with never seen features depending on the loss of a new training instance	Single-label	✓	Zhang et al. (2015)
Rough set-based				
IAMRCD and IARCD	Matrix and non-matrix rough set-based FSS proposals that incrementally update the knowledge granularity when new data and features are received simultaneously	Multi-class	✗	Jing et al. (2018)

$$\mathbf{w}^{t+1} = (\mathbf{w}^t + \tau^t y_t \tilde{\mathbf{x}}_t, \tau^t y_t \hat{\mathbf{x}}_t), \tag{10}$$

where $\tau^t = l^t / \left\|\mathbf{x}_t\right\|^2$. If the instance is correctly classified, i.e., $l^t = 0$, then STSD considers that the new features are not currently necessary and that the weights should not be modified. We can see in Eq. (10) that $\mathbf{w}^{t+1}$ would be an augmentation of $\mathbf{w}^t$ with a weight vector $\hat{\mathbf{w}}^t = \tau^t y_t \hat{\mathbf{x}}_t \in \mathbb{R}^{d^t - d^{t-1}}$, i.e., $\mathbf{w}^{t+1} = (\mathbf{w}^t, \hat{\mathbf{w}}^t)$, which is empty. Notably, the authors also proposed two variants of STSD that modify the computation of τ_t to make the classifier less sensitive to noise. Once the weights are updated, redundant features are removed by selecting a predefined proportion b of features. This process is implemented by projecting the weights to an L_1-ball and then truncating to zero those weights whose values are not in the highest b percent.

Another attempt to perform FSS in this environment is found in Jing et al. (2018), where the *incremental algorithm based on matrix reduction computation for dynamic data mining* (IAMRCD) and the *incremental reduction algorithm for reduct calculation of dynamic data set* (IARCD) are proposed. These rough set-based proposals are able to generate reducts when features and instances vary simultaneously, with the main difference being that IAMRCD is a matrix-based approach, while IARCD is a non-matrix method. This research appears to be based on Jing et al. (2016), who compared the performance of matrix- and non-matrix-based algorithms under feature streams. As already proven in that article, matrix-based approaches for incremental feature reduction can be useful when processing different types of data. However, they are only good at handling small datasets since the run time is really affected by their size. Thus, the IARCD algorithm, whose run times are shorter in the performed experiments, was also proposed. The authors follow the same strategy as that of the MIRA and IARC algorithms (see Sect. 4.2.1), with the difference that the equations used to calculate the incremental knowledge granularity are adapted to accept a variation of both features and instances.

Note that the aforementioned algorithms were designed for supervised classification problems that do not consider any group structure when several new features are received. A summary of them is given in Table 8.

6 Open issues

FSS for streams of data and features is a relatively new branch of study, and the continuous growth of information requires constant improvement. This section discusses some possible directions for future work.

1. *Simplicity of data stream FSS algorithms*. The discussed online algorithms for data streams seek a simple solution to be efficient for real-time problems. However, this can make them of interest for only a small set of problems. Therefore, more research is expected in this area.
2. *Semi-supervised learning and infinitely delayed labels*. Several supervised and unsupervised algorithms were discussed. However, no proposal that focuses on semi-supervised learning was identified. One solution could be to use spectral feature selection in an incremental manner, where the similarity measure between instances considers that class information may or may not exist in some pairwise comparisons. Furthermore, it could be useful to perform FSS on data streams with infinitely delayed labels, i.e., supervised problems where the availability of class variables is delayed. This context was already considered for the classification of data streams (Souza et al. 2015a, b). Unsupervised approaches could be used to perform FSS until the class variable is available, when a supervised method could improve the current feature subset.
3. *Online FSS in other multi-task settings*. Online FSS algorithms could be extended to other multi-task settings not covered by DA-MTFS. For example, there are problems where there is no common set of features due to outlier tasks (Gong et al. 2012).
4. *Past instance removal in rough set-based algorithms*. The majority of works based on rough set theory do not implement a mechanism for the removal of past instances. That is why the possibility of including time windows is advocated in works such as Yang et al. (2018).
5. *Online FSS in multi-dimensional classification problems*. Existing algorithms focus mainly on one-dimensional problems, while few multi-label approaches have been reported. To the best of our knowledge, FSS on multi-dimensional data streams remains an open issue, as current approaches simply address the problem by transforming the data into a one-dimensional learning task. FSS algorithms for feature streams were proposed for multi-label learning but not for the more general multi-dimensional case.
6. *Removal of relevant features by an external agent*. It may be interesting to research the possibility that a feature considered relevant is removed by an external agent (e.g. if it was detected as erroneous) during the processing of a feature stream. An efficient strategy to recover previously removed features using constant memory could be of interest in this situation.
7. *Online ensemble FSS*. Ensemble FSS approaches have received some attention for batch environments (Bolón-Canedo and Alonso-Betanzos 2019), but very little progress has been made for incremental/online problems. For example, to the best of our knowledge, no ensemble proposal exists for online FSS on feature streams. This could be understandable due to the sophistication of some algorithms, which may be unhelpful for real-time data processing. Following an ensemble strategy to define a combined feature selection though different FSS algorithms could lead to inefficient approaches to process dynamic data. However, its robustness could help to obtain more stable solutions.
8. *Algorithms to handle both data and feature streams*. Only two proposals that can handle streams of data and features simultaneously were found. However, this kind of

algorithm is of special interest since they combine the benefits of algorithms for data streams and feature streams into a unique model. When working in study areas that involve, for example, lab experiments, it is common that data not only grow in size but also in dimensionality. Thus, there is a clear absence of research on this topic, which lacks extensions of the classical rough set-based theory to handle, among others, hybrid attributes, apart from algorithms that are not linked to these mathematical theories and deal with, for example, unsupervised, regression, multi-class, multi-dimensional or non-linear learning problems.

9. *Distributed online FSS algorithms*. As the search for more efficient algorithms will continue to be an open issue, we believe that this trend makes it interesting to pay more attention to distributed online FSS. Centralized algorithms may not meet the necessary performance requirements for multiple real-world problems. A starting point could be the development of online FSS methods based on nature-inspired metaheuristics, such as genetic algorithms.

7 Conclusions

Incremental algorithms are characterized by their ability to adapt to the appearance of new data, which forces them to make a strict reduction of consumed resources to address real-time applications. Here, tasks such as dimensionality reduction and, more specifically, feature subset selection (FSS) are implemented due to the inefficiency of maintaining irrelevant and redundant features. Incremental FSS is not only an important task when new features or instances are received over time but also when static datasets are massive since batch algorithms may not be efficient enough to process them. Thus, this review studies a variety of FSS proposals from two main perspectives, depending on whether they are capable of incrementally adapting their solution to data or feature streams. Additionally, a more complex environment where both instances and features are received is introduced.

Although an ideal online FSS algorithm should analyze the data once, some literature methods need to re-examine and store past information, some with the justification of, for example, being able to correctly adapt to concept drift. It is common to find these algorithms defined simply as incremental. Therefore, proposals described as incremental and online are analyzed in this review. Note that there is some ambiguity in the literature about the differences between these concepts. Some works define online learning algorithms as those capable of working in a streaming context endlessly, which seems reasonable, but the classification of an algorithm may depend on the characteristics of the problem being addressed under this definition. A proposal could be considered online if it is able to handle a specific real-time problem but may not be adequate for a more demanding context or when there are, theoretically, an infinite number of incoming relevant and non-redundant features. Consequently, we propose a thorough study of what should be considered as online and incremental FSS when working with data and feature streams, probably using different definitions depending on the type of stream being processed.

There is no unique way to perform FSS, but the definition of a variable as relevant or redundant is dependent on the problem we are facing. Therefore, the FSS task was studied for supervised (from one-dimensional to multi-label) and unsupervised stream learning, as well as for more specific problems, such as multi-task learning, data with hybrid attributes or even environments where instances or features could be removed as they were detected

as erroneous. In addition, the interest in providing as few hyperparameters as possible, since the whole dataset is unknown, has led us to dedicate several sections to the study of rough set-based algorithms and extensions of this theory.

We believe that some of the presented algorithms should be studied in greater depth. The experiments performed in the respective articles may not be sufficiently exhaustive to draw some of their conclusions. Trivial experiments, such as comparing the performance of models built with the entire feature set and the reported subset, are missing in certain works. Furthermore, the strategies followed by each of the algorithms can make them more suitable for certain types of problems, so it is normal that authors focus on highlighting the advantages of their proposals over others. Thus, it is not easy to opt for a single solution from all the proposals since it may depend on the characteristics of the problem being considered and the concessions in terms of efficiency or effectiveness that we are willing to make.

Through the review of the literature, it was found that the considerable attention the field of big data and real-time systems is receiving has led to an increase in research on incremental/online algorithms for streams of new instances and features. This article focuses on 26 different incremental FSS proposals that were published in the past five years, accounting for almost 60% of all analyzed proposals. This trend is expected to increase in the following years since more efficient strategies will be needed to address growing data in terms of the number of instances and dimensionality.

Acknowledgements This work has been partially supported by the Spanish Ministry of Economy, Industry and Competitiveness through the TIN2016-79684-P project and by the Spanish Centre for the Development of Industrial Technology through the IDI-20180156—LearnIIoT project in partnership with Etxetar and Aingura IIoT. C. Villa-Blanco is supported by a predoctoral contract for the formation of doctors from the Universidad Politécnica de Madrid.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

References

Almusallam N, Tari Z, Chan J, AlHarthi A (2018) UFSSF—an efficient unsupervised feature selection for streaming features. In: Proceedings of the 22nd Pacific-Asia conference on knowledge discovery and data mining. Springer, PAKDD'18, pp 495–507. https://doi.org/10.1007/978-3-319-93037-4_39

AlNuaimi N, Masud MM, Serhani MA, Zaki N (2020) Streaming feature selection algorithms for big data: a survey. Appl Comput Inform. https://doi.org/10.1016/j.aci.2019.01.001

Barddal JP, Gomes HM, Enembreck F, Pfahringer B (2017) A survey on feature drift adaptation: definition, benchmark, challenges and future directions. J Syst Softw 127:278–294. https://doi.org/10.1016/j.jss.2016.07.005

BenSaid F, Alimi AM (2021) Online feature selection system for big data classification based on multi-objective automated negotiation. Pattern Recogn 110:107629. https://doi.org/10.1016/j.patcog.2020.107629

Bifet A, Gavaldà R (2009) Adaptive learning from evolving data streams. In: 8th international symposium on intelligent data analysis. Springer, IDA'09, pp 249–260. https://doi.org/10.1007/978-3-642-03915-7_22

Bolón-Canedo V, Alonso-Betanzos A (2019) Ensembles for feature selection: a review and future trends. Inf Fusion 52:1–12. https://doi.org/10.1016/j.inffus.2018.11.008

Boulesnane A, Meshoul S (2017) WD2O: a novel wind driven dynamic optimization approach with effective change detection. Appl Intell 47(2):488–504. https://doi.org/10.1007/s10489-017-0895-2

Boulesnane A, Meshoul S (2018) Effective streaming evolutionary feature selection using dynamic optimization. In: Proceedings of the 6th IFIP international conference on computational intelligence and its applications. Springer, CIIA'18, pp 329–340. https://doi.org/10.1007/978-3-319-89743-1_29

Cai J, Luo J, Wang S, Yang S (2018) Feature selection in machine learning: a new perspective. Neurocomputing 300:70–79. https://doi.org/10.1016/j.neucom.2017.11.077

Chen Z, Liu B (2018) Lifelong machine learning, vol 12. Morgan & Claypool Publishers, San Rafael. https://doi.org/10.2200/s00832ed1v01y201802aim037

Chen D, Yang Y, Dong Z (2016) An incremental algorithm for attribute reduction with variable precision rough sets. Appl Soft Comput 45:129–149. https://doi.org/10.1016/j.asoc.2016.04.003

Cornelis C, Cock MD, Radzikowska AM (2008) Fuzzy rough sets: from theory into practice. Handbook of granular computing. Wiley, New York, pp 533–552. https://doi.org/10.1002/9780470724163.ch24

Das A, Islam MM (2011) SecuredTrust: a dynamic trust computation model for secured communication in multiagent systems. IEEE Trans Dependable Secur Comput 9(2):261–274. https://doi.org/10.1109/TDSC.2011.57

Dhillon PS, Foster D, Ungar L (2010) Feature selection using multiple streams. In: Proceedings of the 13th international conference on artificial intelligence and statistics, PMLR, AISTATS'10, pp 153–160

Diao R, Shen Q (2010) Two new approaches to feature selection with harmony search. In: Proceedings of the 19th international conference on fuzzy systems. IEEE, FUZZ-IEEE'10, pp 1–7. https://doi.org/10.1109/FUZZY.2010.5584009

Diao R, Shen Q (2012) Feature selection with harmony search. IEEE Trans Syst Man Cybern Part B 42(6):1509–1523. https://doi.org/10.1109/TSMCB.2012.2193613

Diao R, Parthaláin NM, Shen Q (2013) Dynamic feature selection with fuzzy-rough sets. In: Proceedings of the 22nd international conference on fuzzy systems. IEEE, FUZZ-IEEE'13, pp 1–7. https://doi.org/10.1109/FUZZ-IEEE.2013.6622410

Ding W, Stepinski TF, Mu Y, Bandeira L, Ricardo R, Wu Y, Lu Z, Cao T, Wu X (2011) Subkilometer crater discovery with boosting and transfer learning. ACM Trans Intell Syst Technol 2(4):39:1-39:22. https://doi.org/10.1145/1989734.1989743

Ditzler G, LaBarck J, Ritchie J, Rosen G, Polikar R (2017) Extensions to online feature selection using bagging and boosting. IEEE Trans Neural Netw Learn Syst 29(9):4504–4509. https://doi.org/10.1109/TNNLS.2017.2746107

Domingos P, Hulten G (2000) Mining high-speed data streams. In: Proceedings of the 6th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, KDD'00, pp 71–80. https://doi.org/10.1145/347090.347107

Dredze M, Crammer K (2008) Online methods for multi-domain learning and adaptation. In: Proceedings of the 13th conference on empirical methods in natural language processing, association for computational linguistics, EMNLP'08, pp 689–697. https://doi.org/10.3115/1613715.1613801

Dredze M, Crammer K, Pereira F (2008) Confidence-weighted linear classification. In: Proceedings of the 25th international conference on machine learning. ACM, ICML'08, pp 264–271. https://doi.org/10.1145/1390156.1390190

Dubois D, Prade H (1990) Rough fuzzy sets and fuzzy rough sets. Int J Gen Syst 17(2–3):191–209. https://doi.org/10.1080/03081079008935107

Elisseeff A, Weston J (2001) A kernel method for multi-labelled classification. In: Proceedings of the 14th International conference on neural information processing systems: natural and synthetic. MIT Press, NIPS'01, pp 681–687

Eskandari S, Javidi MM (2016) Online streaming feature selection using rough sets. Int J Approx Reason 69:35–57. https://doi.org/10.1016/j.ijar.2015.11.006

Fong S, Wong R, Vasilakos AV (2016) Accelerated PSO swarm search feature selection for data stream mining big data. IEEE Trans Serv Comput 9(1):33–45. https://doi.org/10.1109/TSC.2015.2439695

Gama J, Fernandes R, Rocha R (2006) Decision trees for mining data streams. Intell Data Anal 10(1):23–45. https://doi.org/10.3233/ida-2006-10103

Gama J, Žliobait e I, Bifet A, Pechenizkiy M, Bouchachia A (2014) A survey on concept drift adaptation. ACM Comput Surv 46(4):44:1-44:37. https://doi.org/10.1145/2523813

Glocer K, Eads D, Theiler J (2005) Online feature selection for pixel classification. In: Proceedings of the 22nd international conference on machine learning. ACM, ICML'05, pp 249–256. https://doi.org/10.1145/1102351.1102383

Gong P, Ye J, Zhang C (2012) Robust multi-task feature learning. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, KDD'12, pp 895–903. https://doi.org/10.1145/2339530.2339672

Gonzalez-Lopez J, Ventura S, Cano A (2020) Distributed multi-label feature selection using individual mutual information measures. Knowl-Based Syst 188:105052. https://doi.org/10.1016/j.knosys.2019.105052

Gu S, Cheng R, Jin Y (2018) Feature selection for high-dimensional classification using a competitive swarm optimizer. Soft Comput 22(3):811–822. https://doi.org/10.1007/s00500-016-2385-6

Guan L (2009) An incremental updating algorithm of attribute reduction set in decision tables. In: Proceedings of the 6th international conference on fuzzy systems and knowledge discovery. IEEE, FSKD'09, pp 421–425. https://doi.org/10.1109/FSKD.2009.763

Guyon I, Gunn S, Nikravesh M, Zadeh LA (2008) Feature extraction: foundations and applications, vol 207. Springer, New York. https://doi.org/10.1007/978-3-540-35488-8

Hu F, Wang G, Huang H, Wu Y (2005) Incremental attribute reduction based on elementary sets. In: Proceedings of the 10th international conference on rough sets, fuzzy sets, data mining, and granular-soft computing. Springer, RSFDGrC'05, pp 185–193. https://doi.org/10.1007/11548669_20

Hu Q, Yu D, Liu J, Wu C (2008) Neighborhood rough set based heterogeneous feature subset selection. Inf Sci 178(18):3577–3594. https://doi.org/10.1016/j.ins.2008.05.024

Hu X, Zhou P, Li P, Wang J, Wu X (2016) A survey on online feature selection with streaming features. Front Comput Sci 12(3):479–493. https://doi.org/10.1007/s11704-016-5489-3

Huang H, Yoo S, Kasiviswanathan SP (2015) Unsupervised feature selection on data streams. In: Proceedings of the 24th ACM international conference on information and knowledge management. ACM, CIKM'15, pp 1031–1040. https://doi.org/10.1145/2806416.2806521

Hulten G, Spencer L, Domingos P (2001) Mining time-changing data streams. In: Proceedings of the 7th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, KDD'01, pp 97–106. https://doi.org/10.1145/502512.502529

Javidi MM, Eskandari S (2018) Streamwise feature selection: a rough set method. Int J Mach Learn Cybern 9(4):667–676. https://doi.org/10.1007/s13042-016-0595-y

Javidi MM, Eskandari S (2019) Online streaming feature selection: a minimum redundancy, maximum significance approach. Pattern Anal Appl 22(3):949–963. https://doi.org/10.1007/s10044-018-0690-7

Jing Y, Li T, Huang J, Zhang Y (2016) An incremental attribute reduction approach based on knowledge granularity under the attribute generalization. Int J Approx Reason 76:80–95. https://doi.org/10.1016/j.ijar.2016.05.001

Jing Y, Li T, Fujita H, Yu Z, Wang B (2017) An incremental attribute reduction approach based on knowledge granularity with a multi-granulation view. Inf Sci 411:23–38. https://doi.org/10.1016/j.ins.2017.05.003

Jing Y, Li T, Fujita H, Wang B, Cheng N (2018) An incremental attribute reduction method for dynamic data mining. Inf Sci 465:202–218. https://doi.org/10.1016/j.ins.2018.07.001

John GH, Kohavi R, Pfleger K (1994) Irrelevant features and the subset selection problem. In: Proceedings of the 11th international conference on machine learning. Morgan Kaufmann, ICML'94, pp 121–129. https://doi.org/10.1016/B978-1-55860-335-6.50023-4

Kira K, Rendell LA (1992) A practical approach to feature selection. In: Proceedings of the 9th international workshop on machine learning. Morgan Kaufmann, ML'92, pp 249–256. https://doi.org/10.1016/B978-1-55860-247-2.50037-1

Klawonn F, Angelov P (2006) Evolving extended Naive Bayes classifiers. In: Proceedings of the 6th IEEE international conference on data mining-workshops. IEEE, ICDMW'06, pp 643–647. https://doi.org/10.1109/ICDMW.2006.74

Kononenko I (1994) Estimating attributes: analysis and extensions of RELIEF. In: Proceedings of the 7th european conference on machine learning. Springer, ECML'94, pp 171–182. https://doi.org/10.1007/3-540-57868-4_57

Leardi R, Boggia R, Terrile M (1992) Genetic algorithms as a strategy for feature selection. J Chemom 6(5):267–281. https://doi.org/10.1002/cem.1180060506

Li H, Wu X, Li Z, Ding W (2013) Group feature selection with streaming features. In: Proceedings of the 13th IEEE international conference on data mining. IEEE, ICDM'13, pp 1109–1114. https://doi.org/10.1109/ICDM.2013.137

Li J, Hu X, Tang J, Liu H (2015) Unsupervised streaming feature selection in social media. In: Proceedings of the 24th ACM international conference on information and knowledge management. ACM, CIKM'15, pp 1041–1050. https://doi.org/10.1145/2806416.2806501
Li H, Li D, Zhai Y, Wang S, Zhang J (2016) A novel attribute reduction approach for multi-label data based on rough set theory. Inf Sci 367–368:827–847. https://doi.org/10.1016/j.ins.2016.07.008
Li J, Cheng K, Wang S, Morstatter F, Trevino RP, Tang J, Liu H (2017) Feature selection: a data perspective. ACM Comput Surv 50(6):94:1-94:45. https://doi.org/10.1145/3136625
Liang J, Chin KS, Dang C, Yam RCM (2002) A new method for measuring uncertainty and fuzziness in rough set theory. Int J Gen Syst 31(4):331–342. https://doi.org/10.1080/0308107021000013635
Liang J, Wang F, Dang C, Qian Y (2012) An efficient rough feature selection algorithm with a multigranulation view. Int J Approx Reason 53(6):912–926. https://doi.org/10.1016/j.ijar.2012.02.004
Liang J, Wang F, Dang C, Qian Y (2014) A group incremental approach to feature selection applying rough set technique. IEEE Trans Knowl Data Eng 26(2):294–308. https://doi.org/10.1109/TKDE.2012.146
Liberty E (2013) Simple and deterministic matrix sketching. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, KDD'13, pp 581–588. https://doi.org/10.1145/2487575.2487623
Lin Y, Hu Q, Liu J, Li J, Wu X (2017) Streaming feature selection for multilabel learning based on fuzzy mutual information. IEEE Trans Fuzzy Syst 25(6):1491–1507. https://doi.org/10.1109/TFUZZ.2017.2735947
Liu J, Lin Y, Li Y, Weng W, Wu S (2018a) Online multi-label streaming feature selection based on neighborhood rough set. Pattern Recogn 84:273–287. https://doi.org/10.1016/j.patcog.2018.07.021
Liu J, Lin Y, Wu S, Wang C (2018b) Online multi-label group feature selection. Knowl-Based Syst 143:42–57. https://doi.org/10.1016/j.knosys.2017.12.008
Losing V, Hammer B, Wersing H (2018) Incremental on-line learning: a review and comparison of state of the art algorithms. Neurocomputing 275:1261–1274. https://doi.org/10.1016/j.neucom.2017.06.084
Ma Y, Tang J, Aggarwal C (2018) Feature engineering for data streams. Feature engineering for machine learning and data analytics. CRC Press, Boca Raton, pp 117–143
Nguyen HL, Woon YK, Ng WK, Wan L (2012) Heterogeneous ensemble for feature drifts in data streams. In: Proceedings of the 16th Pacific-Asia conference on knowledge discovery and data mining. Springer, PAKDD'12, pp 1–12. https://doi.org/10.1007/978-3-642-30220-6_1
Oh IS, Lee JS, Moon BR (2004) Hybrid genetic algorithms for feature selection. IEEE Trans Pattern Anal Mach Intell 26(11):1424–1437. https://doi.org/10.1109/TPAMI.2004.105
Oza NC, Russell SJ (2001) Online bagging and boosting. In: International workshop on artificial intelligence and statistics, PMLR, pp 229–236
Pawlak Z (1982) Rough sets. Int J Comput Inf Sci 11(5):341–356. https://doi.org/10.1007/BF01001956
Pawlak Z (1997) Vagueness—a rough set view. In: Structures in logic and computer science: a selection of essays in honor of A. Ehrenfeucht. Springer, pp 106–117. https://doi.org/10.1007/3-540-63246-8_7
Pereira RB, Plastino A, Zadrozny B, Merschmann LH (2018) Categorizing feature selection methods for multi-label classification. Artif Intell Rev 49(1):57–78. https://doi.org/10.1007/s10462-016-9516-4
Perkins S, Theiler J (2003) Online feature selection using grafting. In: Proceedings of the 20th international conference on machine learning. AAAI Press, ICML'03, pp 592–599
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
Pintas JT, Fernandes LA, Garcia ACB (2021) Feature selection methods for text classification: a systematic literature review. Artif Intell Rev 54(8):6149–6200. https://doi.org/10.1007/s10462-021-09970-6
Pudil P, Novovičová J, Kittler J (1994) Floating search methods in feature selection. Pattern Recogn Lett 15(11):1119–1125. https://doi.org/10.1016/0167-8655(94)90127-9
Qian Y, Liang J (2008) Combination entropy and combination granulation in rough set theory. Int J Uncertain Fuzziness Knowl-Based Syst 16(2):179–193. https://doi.org/10.1142/S0218488508005121
Rahmaninia M, Moradi P (2018) OSFSMI: online stream feature selection method based on mutual information. Appl Soft Comput 68:733–746. https://doi.org/10.1016/j.asoc.2017.08.034
Saeys Y, Inza I, Larrañaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23(19):2507–2517. https://doi.org/10.1093/bioinformatics/btm344
Salperwyck C, Lemaire V, Hue C (2015) Incremental weighted Naive Bayes classifiers for data stream. In: Data science, learning by latent structures, and knowledge discovery. Springer, pp 179–190. https://doi.org/10.1007/978-3-662-44983-7_16
Shao W, He L, Lu CT, Wei X, Yu PS (2016) Online unsupervised multi-view feature selection. In: Proceedings of the 16th IEEE international conference on data mining. IEEE, ICDM'16, pp 1203–1208. https://doi.org/10.1109/ICDM.2016.0160

Shu W, Qian W (2015) An incremental approach to attribute reduction from dynamic incomplete decision systems in rough set theory. Data Knowl Eng 100:116–132. https://doi.org/10.1016/j.datak.2015.06.009
Shu W, Qian W, Xie Y (2019) Incremental approaches for feature selection from dynamic data with the variation of multiple objects. Knowl-Based Syst 163:320–331. https://doi.org/10.1016/j.knosys.2018.08.028
Skowron A, Rauszer C (1992) The discernibility matrices and functions in information systems. In: Intelligent decision support: handbook of applications and advances of the rough sets theory. Springer, chap 2:3, pp 331–362. https://doi.org/10.1007/978-94-015-7975-9_21
Solorio-Fernández S, Ochoa JAC, Martínez-Trinidad JF (2020) A review of unsupervised feature selection methods. Artif Intell Rev 53:907–948. https://doi.org/10.1007/s10462-019-09682-y
Somasundaram GD, Mylsamy S (2018) Feature selection, online feature selection techniques for big data classification: a review. In: Proceedings of the 1st international conference on current trends towards converging technologies. IEEE, ICCTCT'18, pp 1–9. https://doi.org/10.1109/ICCTCT.2018.8550928
Souza VMA, Silva DF, Batista GEAPA, Gama J (2015a) Classification of evolving data streams with infinitely delayed labels. In: Proceedings of the 14th international conference on machine learning and applications. IEEE, ICMLA'15, pp 214–219. https://doi.org/10.1109/ICMLA.2015.174
Souza VMA, Silva DF, Gama J, Batista GEAPA (2015b) Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In: Proceedings of the 15th SIAM international conference on data mining. SIAM, SDM'15, pp 873–881. https://doi.org/10.1137/1.9781611974010.98
Swan M (2012) Health 2050: the realization of personalized medicine through crowdsourcing, the quantified self, and the participatory biocitizen. J Personalized Med 2(3):93–118. https://doi.org/10.3390/jpm2030093
Tommasel A, Godoy D (2016) Short-text feature construction and selection in social media data: a survey. Artif Intell Rev 49(3):301–338. https://doi.org/10.1007/s10462-016-9528-0
Torrey L, Shavlik J (2010) Transfer learning. In: Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. IGI Global, pp 242–264. https://doi.org/10.4018/978-1-60566-766-9.ch011
Trohidis K, Tsoumakas G, Kalliris G, Vlahavas I (2011) Multi-label classification of music by emotion. EURASIP J Audio Speech Music Process 2011:4. https://doi.org/10.1186/1687-4722-2011-426793
Ungar LH, Zhou J, Foster DP, Stine BA (2005) Streaming feature selection using IIC. In: Proceedings of the 10th international workshop on artificial intelligence and statistics, society for artificial intelligence and statistics, AISTATS'05, pp 357–364
Urbanowicz RJ, Meeker M, La Cava W, Olson RS, Moore JH (2018) Relief-based feature selection: introduction and review. J Biomed Inform 85:189–203. https://doi.org/10.1016/j.jbi.2018.07.014
Venkatesh B, Anuradha J (2019) A review of feature selection and its methods. Cybern Inf Technol 19(1):3–26. https://doi.org/10.2478/cait-2019-0001
Wang F, Liang J, Qian Y (2013) Attribute reduction: a dimension incremental strategy. Knowl-Based Syst 39:95–108. https://doi.org/10.1016/j.knosys.2012.10.010
Wang J, Zhao P, Hoi SCH, Jin R (2014) Online feature selection and its applications. IEEE Trans Knowl Data Eng 26(3):698–710. https://doi.org/10.1109/TKDE.2013.32
Wang J, Wang M, Li P, Liu L, Zhao Z, Hu X, Wu X (2015) Online feature selection with group structure analysis. IEEE Trans Knowl Data Eng 27(11):3029–3041. https://doi.org/10.1109/TKDE.2015.2441716
Wang H, Wang G, Zeng X, Peng S (2017) Online streaming feature selection based on conditional information entropy. In: Proceedings of the 8th IEEE international conference on big knowledge. IEEE, ICBK'17, pp 230–235. https://doi.org/10.1109/ICBK.2017.44
Wang H, Yu D, Li Y, Li Z, Wang G (2018) Multi-label online streaming feature selection based on spectral granulation and mutual information. In: Proceedings of the 2018 international joint conference on rough sets. Springer, IJCRS'18, pp 215–228, https://doi.org/10.1007/978-3-319-99368-3_17
Webb GI, Hyde R, Cao H, Nguyen HL, Petitjean F (2016) Characterizing concept drift. Data Min Knowl Disc 30(4):964–994. https://doi.org/10.1007/s10618-015-0448-4
Weinberger K, Dasgupta A, Langford J, Smola A, Attenberg J (2009) Feature hashing for large scale multitask learning. In: Proceedings of the 26th international conference on machine learning. ACM, ICML'09, pp 1113–1120. https://doi.org/10.1145/1553374.1553516
Wierman MJ (1999) Measuring uncertainty in rough set theory. Int J Gen Syst 28(4–5):283–297. https://doi.org/10.1080/03081079908935239
Wu X, Yu K, Ding W, Wang H, Zhu X (2013) Online feature selection with streaming features. IEEE Trans Pattern Anal Mach Intell 35(5):1178–1192. https://doi.org/10.1109/TPAMI.2012.197
Wu Y, Hoi SCH, Mei T, Yu N (2017) Large-scale online feature selection for ultra-high dimensional sparse data. ACM Trans Knowl Discov Data 11(4):48:1-48:22. https://doi.org/10.1145/3070646

Xu Y, Wang L, Zhang R (2011) A dynamic attribute reduction algorithm based on 0–1 integer programming. Knowl-Based Syst 24(8):1341–1347. https://doi.org/10.1016/j.knosys.2011.06.007
Yang H, Lyu MR, King I (2013) Efficient online learning for multitask feature selection. ACM Trans Knowl Discov Data 7(2):6:1-6:27. https://doi.org/10.1145/2499907.2499909
Yang Y, Chen D, Wang H, Wang X (2018) Incremental perspective for feature selection based on fuzzy rough sets. IEEE Trans Fuzzy Syst 26(3):1257–1273. https://doi.org/10.1109/TFUZZ.2017.2718492
Yi BK, Sidiropoulos ND, Johnson T, Jagadish HV, Faloutsos C, Biliris A (2000) Online data mining for co-evolving time sequences. In: Proceedings of the 16th international conference on data engineering. IEEE, ICDE'00, pp 13–22. https://doi.org/10.1109/ICDE.2000.839383
Yu L, Liu H (2003) Feature selection for high-dimensional data: a fast correlation-based filter solution. In: Proceedings of the 20th international conference on machine learning. AAAI Press, ICML'03, pp 856–863
Yu L, Liu H (2004) Efficient feature selection via analysis of relevance and redundancy. J Mach Learn Res 5:1205–1224
Yu K, Wu X, Ding W, Pei J (2014) Towards scalable and accurate online feature selection for big data. In: Proceedings of the 14th IEEE international conference on data mining. IEEE, ICDM'14, pp 660–669. https://doi.org/10.1109/ICDM.2014.63
Yu K, Ding W, Wu X (2016a) LOFS: a library of online streaming feature selection. Knowl-Based Syst 113:1–3. https://doi.org/10.1016/j.knosys.2016.08.026
Yu K, Wu X, Ding W, Pei J (2016b) Scalable and accurate online feature selection for big data. ACM Trans Knowl Discov Data 11(2):16:1-16:39. https://doi.org/10.1145/2976744
Zeng A, Li T, Liu D, Zhang J, Chen H (2015) A fuzzy rough set approach for incremental feature selection on hybrid information systems. Fuzzy Sets Syst 258:39–60. https://doi.org/10.1016/j.fss.2014.08.014
Zhang Q, Zhang P, Long G, Ding W, Zhang C, Wu X (2015) Towards mining trapezoidal data streams. In: Proceedings of the 15th IEEE international conference on data mining. IEEE, ICDM'15, pp 1111–1116. https://doi.org/10.1109/ICDM.2015.42
Zhang Q, Xie Q, Wang G (2016) A survey on rough set theory and its applications. CAAI Trans Intell Technol 1(4):323–333. https://doi.org/10.1016/j.trit.2016.11.001
Zhang R, Nie F, Li X, Wei X (2019) Feature selection with multi-view data: a survey. Inf Fusion 50:158–167. https://doi.org/10.1016/j.inffus.2018.11.019
Zhao Z, Liu H (2007) Spectral feature selection for supervised and unsupervised learning. In: Proceedings of the 24th international conference on machine learning. ACM, ICML'07, pp 1151–1157. https://doi.org/10.1145/1273496.1273641
Zhao ZA, Liu H (2011) Spectral feature selection for data mining. Chapman and Hall/CRC, Boca Raton. https://doi.org/10.1201/b11426
Zhao Z, Wang L, Liu H (2010) Efficient spectral feature selection with minimum redundancy. In: Proceedings of the 24th AAAI conference on artificial intelligence. AAAI Press, AAAI'10, pp 673–678
Zhou J, Foster D, Stine R, Ungar L (2005) Streaming feature selection using alpha-investing. In: Proceedings of the 11th ACM SIGKDD international conference on knowledge discovery in data mining. ACM, KDD'05, pp 384–393. https://doi.org/10.1145/1081870.1081914
Zhou P, Hu X, Li P, Wu X (2017) Online feature selection for high-dimensional class-imbalanced data. Knowl-Based Syst 136:187–199. https://doi.org/10.1016/j.knosys.2017.09.006
Zhou P, Hu X, Li P, Wu X (2019a) OFS-density: a novel online streaming feature selection method. Pattern Recogn 86:48–61. https://doi.org/10.1016/j.patcog.2018.08.009
Zhou P, Hu X, Li P, Wu X (2019b) Online streaming feature selection using adapted neighborhood rough set. Inf Sci 481:258–279. https://doi.org/10.1016/j.ins.2018.12.074
Zhu S, Ji X, Xu W, Gong Y (2005) Multi-labelled classification using maximum entropy method. In: Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval. ACM, SIGIR'05, pp 274–281. https://doi.org/10.1145/1076034.1076082
Ziarko W (1993) Variable precision rough set model. J Comput Syst Sci 46(1):39–59. https://doi.org/10.1016/0022-0000(93)90048-2