Expert Systems With Applications 185 (2021) 115589

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Automatic recommendation of feature selection algorithms based on dataset characteristics

Antonio Rafael Sabino Parmezan [a,*], Huei Diana Lee [b], Newton Spolaôr [b], Feng Chung Wu [b,c]

[a] Laboratory of Computational Intelligence, Institute of Mathematics and Computer Science, University of São Paulo, Av. Trabalhador São-carlense 400, ZIP code 13566-590 São Carlos, SP, Brazil

[b] Laboratory of Bioinformatics, Engineering and Exact Sciences Center, Western Paraná State University, Av. Tarquínio Joslin dos Santos 1300, 85867-900 Foz do Iguaçu, PR, Brazil

[c] Coloproctology Service, Faculty of Medical Sciences, University of Campinas, Rua Tessália Vieira de Camargo 126, Cidade Universitária Zeferino Vaz, 13083-887 Campinas, SP, Brazil

ARTICLE INFO

Keywords:
Feature engineering
Characterization measures
Algorithm selection
Recommendation system
Filter
Wrapper

ABSTRACT

Feature selection in real-world data mining problems is essential to make the learning task efficient and more accurate. Identifying the best feature selection algorithm, among the many available, is a complex activity that still relies heavily on human experts or some random trial-and-error procedure. Thus, the automated machine learning community has taken some steps towards the automation of this process. In this paper, we address the metalearning challenge of recommending feature selection algorithms by proposing a novel meta-feature engineering model. Our model considers a broad collection of meta-features that enable the study of the relationship between the dataset properties and the feature selection algorithm performance in terms of several criteria. We arrange the input meta-features into eight categories: (i) simple, (ii) statistical, (iii) information-theoretical, (iv) complexity, (v) landmarking, (vi) based on symbolic models, (vii) based on images, and (viii) based on complex networks (graphs). The target meta-features emerge from a multi-criteria performance measure, based on five individual performance indexes, that assesses feature selection methods grounded in information, distance, dependence, consistency, and precision measures. We evaluate our proposal using a recently developed framework that extracts the input meta-features from 213 benchmark datasets, and ranks the assessed feature selection algorithms, to fill in the target meta-features in meta-bases. This evaluation uses five state-of-the-art classification methods to induce recommendation models from meta-bases: C4.5, Random Forest, XGBoost, ANN, and SVM. The results showed that it is possible to reach an average accuracy of up to 90% applying our meta-feature engineering model. This work is the first to use an extensive empirical evaluation to provide a careful discussion of the strengths and limitations of more than 160 meta-features. These meta-features, while designed to aid the task of feature selection algorithm recommendation, can readily be employed in other metalearning scenarios. Therefore, we believe our findings are a valuable contribution to the fields of automated machine learning and data mining, as well as to the feature extraction and pattern recognition communities.

1. Introduction

Intelligent data analysis in high-dimensional space remains a challenge for experts and practitioners in the fields of data mining and knowledge discovery (Bolón-Canedo et al., 2015; Li et al., 2017; Yu & Liu, 2003). Feature selection arises in this context as an effective way of helping not only to reduce data dimensionality from removing redundant and irrelevant features, but also minimizing computation time to build more accurate and understandable machine learning models (Liu & Motoda, 2007; Yu & Liu, 2004).

For the past two decades, much research effort has been devoted to developing algorithms able to select, in agreement with some criterion or measure of importance, relevant and non-redundant features from large databases (Li et al., 2017). Currently, with the advancement of technology and the maturity achieved by such studies, we have at our disposal a broad collection of candidate algorithms to perform the feature selection task (Brezočnik et al., 2018; Chandrashekar & Sahin, 2014; Das, 2001; Dash & Liu, 1997; Dy & Brodley, 2004; Sheikhpour et al., 2017).

* Corresponding author.
E-mail addresses: parmezan@usp.br (A.R.S. Parmezan), huei.lee@unioeste.br (H.D. Lee), newton.spolaor@unioeste.br (N. Spolaôr), wu.chung@unioeste.br (F.C. Wu).

https://doi.org/10.1016/j.eswa.2021.115589
Received 27 July 2020; Received in revised form 24 June 2021; Accepted 8 July 2021
Available online 13 July 2021

However, given the wide range of feature selection algorithms, some of them with the need for multiple parameter settings, another problem emerges: choosing the most suitable feature selection algorithm for each problem or application domain.

In practical terms, the choice of the most appropriate feature selection algorithm for a particular dataset depends basically on two aspects (Parmezan et al., 2017): the technical knowledge about the feature selection algorithm, dependent on the machine learning experts, and the domain knowledge, in general, dependent on the domain experts.

As the choice of feature selection algorithms is still strongly dependent on human experts, which are rare and expensive, many surveys involving extensive empirical evaluations have been conducted to guide this decision (Arauzo-Azofra et al., 2011; Bolón-Canedo et al., 2013; Molina et al., 2002). If on the one hand, these experimental works provide valuable results from the scientific point of view, on the other, they are extremely expensive (quite time-consuming) since they are based on a random process of trial and error (Brodley, 1993). Moreover, the outcomes of these studies usually relate so many variables and properties that using them consistently is rather complex for a non-expert user.

In order to address the mentioned problem more efficiently, the automated machine learning community recently began to investigate the use of metalearning for the automatic recommendation of feature selection algorithms (Aduviri et al., 2018; Filchenkov & Pendryak, 2015; Goswami et al., 2016; Parmezan et al., 2017; Shilbayeh & Vadera, 2014; Wang et al., 2013). The core idea of this approach is to think of a feature selection algorithm recommendation as a supervised learning problem in which (Lemke et al., 2015): (i) datasets are examples (objects); (ii) a dataset is described through its properties known as input meta-features; (iii) the target function maps a dataset to an algorithm, which presents the best performance on this dataset regarding a particular criterion. Thus, the algorithm selection problem is reduced to learning such target function, which can predict the most appropriate algorithm for a given dataset with a specific set of properties.

The few existing frameworks on metalearning for feature selection algorithm recommendation are based on the sharpened no-free-lunch theorem (Wolpert & Macready, 1997). This theorem states that there is no single algorithm that is always the best for all datasets. In particular, experiments have confirmed that some feature selection algorithms work well for some datasets and perform poorly on others (Arauzo-Azofra et al., 2011; Molina et al., 2002).

We, like many other authors, believe the characteristics of the dataset might have some effect on the algorithm for feature selection. If, on the one hand, this hypothesis encourages us to use metalearning to model the relationship between the datasets' characteristics and the feature selection algorithms' performance, on the other hand, it makes us answer three fundamental issues: (i) which input meta-features are the most promising to characterize a dataset; (ii) how to evaluate the performance of a feature selection algorithm and identify the applicable one(s) for a given dataset; (iii) how to suggest feature selection algorithms for a new dataset.

In this paper, we address the aforementioned challenges by proposing a novel meta-feature engineering model that may be a reference in the feature selection algorithm recommendation via metalearning. Our proposal includes not only an unprecedented collection of dataset characterization measures (input meta-features) but also a set of performance measures for feature selection algorithm evaluation (target meta-features). Thus, this work is the first to evaluate such a broad and diversified meta-feature engineering model. Our study also aims to find an input meta-feature collection that allows discriminating the performance of distinct feature selection algorithms with high reliability and low computational cost. The three major contributions of this work are:

1. We introduce a novel meta-feature engineering model that was specifically designed to support the automatic recommendation of feature selection algorithms efficiently. The input meta-features explored in this article can be arranged into eight categories: (i) simple, (ii) statistical, (iii) information-theoretical, (iv) complexity, (v) landmarking, (vi) based on symbolic models, (vii) based on images, and (viii) based on complex networks (graphs). The target meta-features, in turn, involve five individual performance indexes and one multi-criteria performance measure;
2. We evaluate the proposed meta-feature engineering model using a recently developed framework to rank feature selection algorithms given the dataset characteristics. We can see the recommendation process as a specific application of metalearning, where suggestion models of feature selection algorithms (meta-models) are induced using machine learning classification methods. We assess the approach with a set of 213 benchmark datasets, and five feature selection algorithms configured with distinct search strategies. We adopted the following state-of-the-art classification algorithms for the meta-models construction: C4.5, Random Forest, XGBoost, Artificial Neural Network (ANN), and Support Vector Machine (SVM). As we discussed throughout this paper, when the meta-model has a high predictive performance, it can predict the most promising feature selection algorithm(s);
3. We explore the relationship between input and target meta-features to understand how strongly a set of properties, represented by a dataset, matches the bias of each feature selection algorithm and its configuration. As a result, we have identified an input meta-feature collection, *i.e.* an input meta-feature set with high discriminative power and low computational cost, which can be used by metalearning architectures to recommend feature selection algorithms properly.

It is worth pointing out that our findings, together with our careful discussion of the strengths and limitations of the most prominent investigated meta-features – some of them new in the area – are a valuable contribution to the fields of automated machine learning and data mining, as well as to the feature extraction and pattern recognition communities.

The remainder of this paper is structured as follows: Section 2 provides the fundamentals of feature selection and metalearning. Section 3 details the problem of recommending feature selection algorithms, as well the challenges it poses for data mining systems. Section 4 describes the proposed meta-feature engineering model. Section 5 specifies the experimental setting, while Section 6 presents and discusses the results. Lastly, Section 7 shows the achievements and remaining challenges of our line of research.

2. Background and definitions

This section presents the central concepts and definitions related to feature selection and metalearning. First, we introduce the problem of feature selection as a process of searching for features or optimal subsets of features, according to some criterion of importance. Here, we also discuss the importance measures to evaluate features according to the classical (information, correlation, and distance), consistency and precision categories. Next, we describe the fundamentals of the metalearning area, which aim to help machine learning automation by constructing efficient and effective systems for algorithm recommendation.

2.1. Feature selection

Several methods were proposed for processing data and building models to represent new and significant knowledge, from the performance and the comprehensibility point of view (Han et al., 2011). In this context, one of the simplest and most used forms to represent data is performed through its attributes and their respective values, and is so-called attribute-value table. These attributes, also referred to

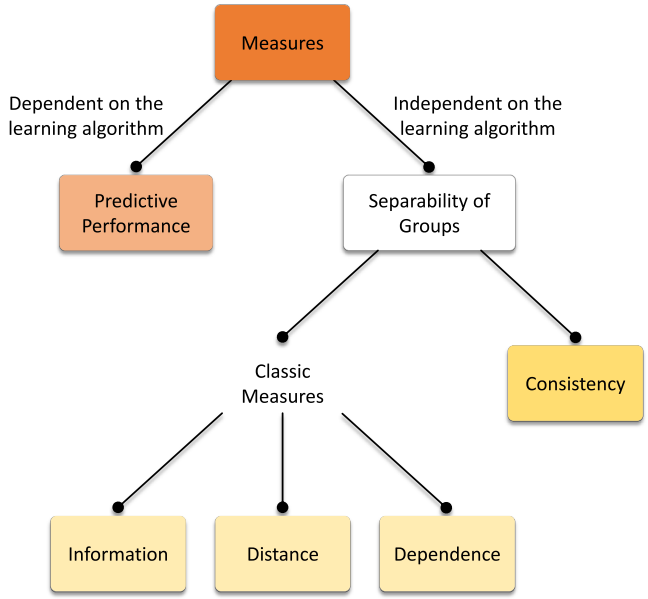

Fig. 1. Hierarchy of importance measures.

as features, variables or characteristics, may assume quantitative and qualitative values, and feed feature transformation and feature selection operations. Feature transformation covers the tasks of constructing, extracting, and discretizing attributes. In turn, feature selection, focused on this work, has as objective to determine an optimal subset of features – Feature Subset Selection (FSS) – or an ordered set of features – ranking –, according to some criterion or measure of importance (Fig. 1). Feature selection output aims to represent the vital information contained in the data (Han et al., 2011; Liu & Motoda, 2007). FSS methods search for minimum subsets of features, usually uncorrelated among each other and relevant to the class, while ranking methods order attributes by individual evaluation. Thus, highly relevant features with the class, but redundant with others, will be on the top of the ranking. As mentioned, both strategies consider some criterion of importance.

Aside from how attributes are evaluated, another important aspect is how the feature selection method relates with the learning algorithm that will be used to further build the model containing patterns from the data. In this sense, three main FSS approaches may be mentioned (Fig. 2).

Although the wrapper and embedded approaches are highly related to the learning algorithm, the filter approach considers no properties nor characteristics of such an algorithm. The idea is to filter non-important attributes according to some criteria that reflect the general characteristics of the datasets. Thus, methods applied according to the filter approach are independent of the learning algorithm, which receives the set of examples as input described using only the subset of important attributes identified by the feature selection task (Blum & Langley, 1997; John et al., 1994).

As for the wrapper approach, the feature selection process also occurs externally to the learning algorithm but uses such an algorithm as a black box to analyze, in terms of performance, at each iteration, the subset of attributes in question. In other words, the candidate subset of attributes is generated at a specific iteration, the model is constructed using the learning algorithm considering such a subset of the training set. Finally, the resulting predictive performance of the induced model is used as a measure of importance for evaluating the subset of attributes investigated. This process is repeated for each subset of attributes until a given stop criterion is satisfied (Kohavi & John, 1997).

Usually, the subsets of attributes selected from the wrapper approach, for a given learning algorithm, tend to result in a better performance in the induced model than the subsets selected through the filter approach. This is because the induction of the model by this specific learning algorithm follows the bias of the same algorithm used during the feature selection process. Thus, the selected attributes are supposed to be optimal, concerning the predictive performance, for models generated by the considered inductor. However, the filter approach has, in general, a lower computational cost than the wrapper strategy, since it does not require the construction of a new model for each subset of attributes to be evaluated (Das, 2001; Liu & Motoda, 2007).

Finally, the embedded approach relates to the learning algorithm in the sense that the feature selection task is performed internally by the pattern extraction algorithm itself. Therefore, methods applied according to this approach select the subset of attributes in the process of building the induction model, during the training phase, and are usually specific to a given learning algorithm (Liu & Motoda, 2007).

2.2. Metalearning

From the perspective of the data mining area, metalearning can be understood as an approach to adapt, based on experience, a learning system dealing with specific data. The experience comes from knowledge associated with previous learning tasks in the same data or data from other domains (Lemke et al., 2015). In turn, the adaptation promotes, for example, comprehension in the way that learning can occur in different domains with flexibility and proper performance.

A typical application in data mining considers the metalearning system as an algorithm selector able to recommend algorithms better suited to some data of interest. The subject inherent to this application can be defined as follows. Let D, A and Y be, respectively, the problem repository (datasets), algorithm, and performance measure spaces. Given the problem $d_i \in D$, described by the features $\mathrm{f}(d_i)$, a metalearning system can be designed to map $\mathrm{f}(d_i)$ into A, in order to automatically select the algorithm $a \in A$ that maximizes the relation $y(a, \mathrm{f}(d_i)) \in Y$.

Metalearning has inspired research in data mining since the 1970s at least (Rice, 1976). The community started to explicitly use the term metalearning in the 1990s. From that moment on, more metalearning definitions appeared in the literature, as indicated in Parmezan et al. (2017) and Lemke et al. (2015), supporting a better understanding of the concept. Several academic projects also emerged, promoting the dissemination of frameworks, architectures, and computational tools. The European project e-LICO and the Auto-WEKA tool,[1] a Weka (Witten et al., 2011) extension, illustrate this fact.

This work uses metalearning to recommend feature selection algorithms to be applied in datasets of interest, as described in Section 3. But metalearning has already been applied for other data mining-related problems, such as the choice of weights for time series forecasting methods (Montero-Manso et al., 2020) and the number of groups in clustering (Lee & Olafsson, 2013), as well as hyperparameter optimization in SVM (Mantovani et al., 2019).

3. Metalearning for feature selection algorithm recommendation

The problem of the proper choice of feature selection algorithms has received increasing attention from the automated machine learning research community, as it represents a challenge in data mining that, if overcome, promises to accelerate the productivity of data scientists and machine learning practitioners (Parmezan et al., 2017).

The need for an automatic recommendation of feature selection methods is due to the existence of a wide range of algorithms developed

[1] https://www.cs.ubc.ca/labs/beta/Projects/autoweka/.

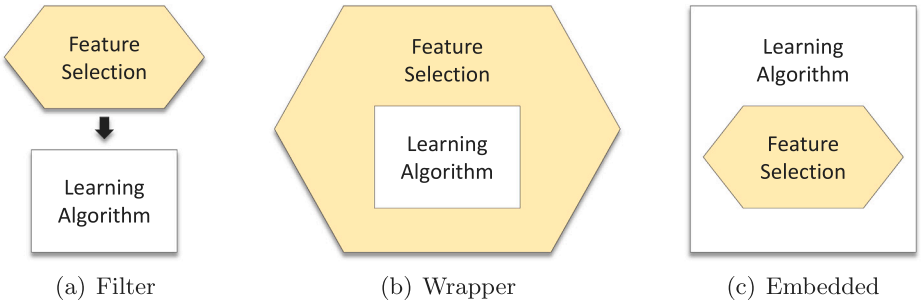

Fig. 2. Approaches for feature subset selection.

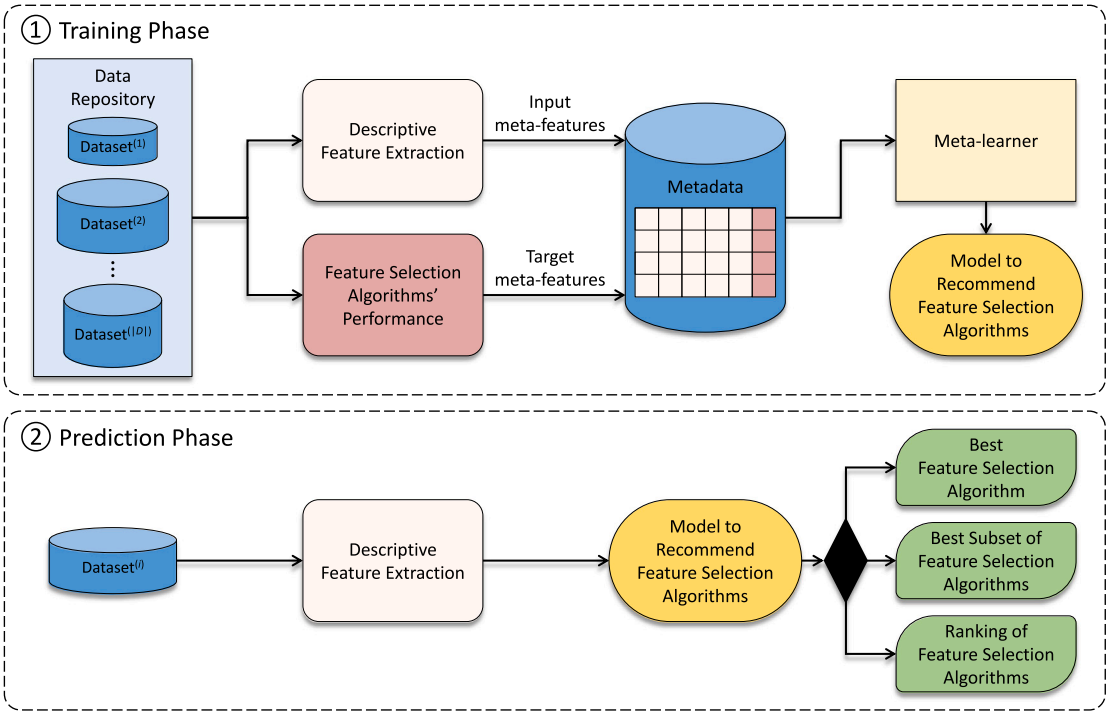

Fig. 3. High-level overview of the metalearning framework for feature selection algorithm recommendation.

with the purpose of selecting relevant and non-redundant features (Brezočnik et al., 2018; Chandrashekar & Sahin, 2014; Dy & Brodley, 2004; Sheikhpour et al., 2017). In most applications or problems, choosing one among all available feature selection methods becomes both a subjective and a costly decision. Subjectivity happens because deciding which feature selection algorithm to use depends not only on the technical details of these methods but also on the data characteristics. Costliness, in turn, includes the time spent by computer and domain experts, as well as exhaustive empirical assessments often used to guide such decisions. The inappropriate choice of feature selection algorithms can lead to severe problems, such as compromising the quality of the patterns to be extracted and, consequently, the result of eventual decision making.

In the last seven years, metalearning fundamentals have been naturally adapted to help automatic recommendation of feature selection methods (Aduviri et al., 2018; Filchenkov & Pendryak, 2015; Parmezan et al., 2017; Shilbayeh & Vadera, 2014; Wang et al., 2013). Although this field of study is still immature, the literature already reports some guidelines that we must follow in conducting this task.

Consider a space of datasets $D = \{d_1, d_2, \ldots, d_{|D|}\}$, a candidate set of feature selection algorithms $A = \{a_1, a_2, \ldots, a_{|A|}\}$, and a certain quality criterion $Q : A \times D \rightarrow \Re$. Suppose we already know a set of meta-features that describe the datasets: $I_1, I_2, \ldots, I_{|D|}$, $I_i : D \rightarrow$ Codomain(I_i). We also have at our disposal a meta-base containing $|D|$ pairs of meta-examples (I_i, T_i), where $T_i \subseteq A$ and represents the concept to be learned. The goal is to apply a supervised machine learning method on the meta-base to induce a recommendation model of feature selection algorithms (meta-model).

Given the meta-features of a new dataset d_i, we can use the meta-model to suggest to d_i the most promising feature selection method(s) T_i. Fig. 3 provides a high-level overview of the architecture that aims to help the user in the process of choosing the most appropriate feature selection algorithm(s) given a new unseen dataset instance.

Fig. 3 organizes the structure of the recommendation process of feature selection methods in two phases. In the training phase, from each historical dataset belonging to a sufficiently large data repository, we must first extract descriptive characteristics — so-called input meta-features. Then, we need to empirically evaluate the available feature selection algorithms on the historical datasets to obtain information about their performance. With these performances at hand, we can rank the feature selection methods from best to worst for each dataset from the repository. A ranking of feature selection algorithms – ordered list of discrete or numeric labels – is called a target meta-feature. Input meta-features associated with target meta-features result in meta-examples that make up a meta-base (metadata). From now on, it is possible to apply over the meta-base a supervised machine learning

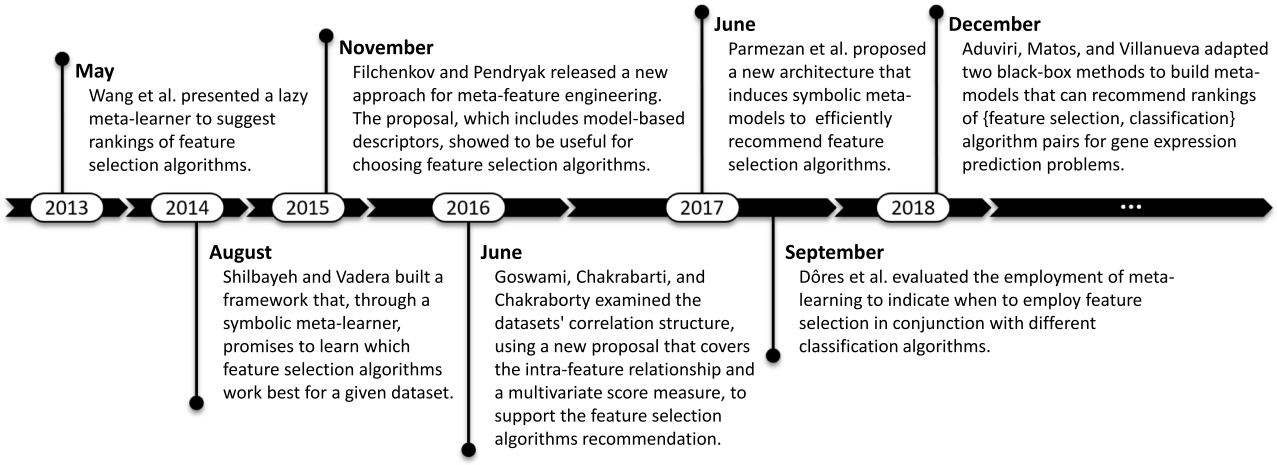

Fig. 4. Timeline of publications in the field of metalearning for feature selection algorithm recommendation.

method – a metalearner that can be a regressor or a classifier – to automatically acquire knowledge, associating the dataset characteristics with the feature selection algorithms' performance. In practical terms, the metalearner generates a meta-model that maps the metaknowledge embedded in the metadata and will be employed in the next phase to guide the choice of feature selection methods.

In the testing phase, the meta-model can then be used to efficiently predict the most appropriate feature selection algorithm(s) for a given new dataset. We must emphasize that there are three distinct approaches to suggesting methods for user appreciation (Kalousis, 2002). The first approach seeks to provide the best algorithm, *i.e.*, the one that supposedly produces the best result for a given task or dataset. The second approach indicates, among the candidate methods, the collection of algorithms that presented the best performance estimation for the dataset. This collection contains the algorithms that do not statistically outperform the best one, besides the best method itself. The third approach, namely ranking, displays the algorithms in order of preference regarding the dataset. The ordering criterion can be both the value of an individual performance index (Aduviri et al., 2018; Filchenkov & Pendryak, 2015; Shilbayeh & Vadera, 2014) and the result of a multi-criteria performance measure (Parmezan et al., 2017; Wang et al., 2013).

The way to suggest feature selection methods has a direct impact on the metalearning architecture as it determines how target meta-features – which and how many feature selection algorithms – should be linked with input meta-features to compose the training meta-base.

In the next subsections, we describe the state-of-the-art in feature selection method recommendation via metalearning. We begin by reporting related work, which covers metalearning frameworks with distinct settings. Afterward, we present the dataset characterization measures that were used to extract input meta-features. Finally, we discuss the main performance evaluation measures of feature selection algorithms for target meta-feature construction.

3.1. Previous work

As feature selection algorithm recommendation is practically useful in any data analysis scenario, the development of systems that automate this task is an emerging challenge in the metalearning field. The fundamental issues to be considered in designing such systems are: (i) which input meta-features are the most promising to characterize a dataset; (ii) how to evaluate the performance of a feature selection algorithm and identify the applicable one(s) for a dataset of interest; (iii) how to suggest feature selection algorithms for a new dataset.

The problem regarding the recommendation of feature selection algorithms using metalearning is relatively new, and few studies offer meaningful advances towards solving it. Fig. 4 shows a timeline of papers written in English that have been published on metalearning for feature selection algorithm recommendation since 2013.

Fig. 4 shows that, from January 2013 to December 2016, only one paper was published annually. However, interest in the research theme grew in 2017 and now appears to become a hot study topic. Most of these studies involve the proposition of metalearning frameworks to suggest feature selection algorithms. Each proposed architecture involves different dataset characterization measures that can be seen as parameters of the frameworks. At this point, we need to note that each paper recommends a subset of the most promising input meta-features.

The feature selection algorithms investigated by previous work are diverse, and some of these studies include even the analysis of the search strategy and direction (Filchenkov & Pendryak, 2015; Wang et al., 2013). In this way, each combination – a feature selection method with a search strategy and direction – is typically evaluated according to the predictive performance of a model built from the subset of features selected by the combination, as well the percentage of reduction of attributes and the learning time for the building of such model. Below, we describe related work in more detail.

Wang et al. (2013) introduced a lazy method for automatic feature selection algorithm suggestion. The method identifies, with the help of a distance measure and employing the k-Nearest Neighbors (kNN) search, the k most similar datasets to a given new dataset. Then, using a multi-criteria metrics, it estimates the performance of all candidate feature selection algorithms on the similar k datasets and, based on the estimated performances, orders the r best feature selection algorithms. The researchers built a training meta-base from 115 datasets described by 13 characterization measures belonging to the simple, statistical, and information categories. Four feature selection algorithms, each with different strategies and search directions, were used to evaluate the method. The results indicated the proposal's effectiveness for feature selection algorithm recommendation.

Shilbayeh and Vadera (2014) designed a metalearning framework that can learn about which feature selection algorithms work best on a dataset of interest. The authors implemented and tested part of this framework using a meta-base that represents 26 datasets described by six characterization measures — three from the simple category and three from the information category. The researchers linked the extracted features to three feature selection approaches under different search techniques, along with applying six classifiers and fed them to a symbolic inductor to create a meta-model. The results overview showed that the derived metaknowledge seems to be useful.

Filchenkov and Pendryak (2015) presented a new meta-feature engineering approach, which demonstrated to be prominent for the automatic suggestion of feature selection algorithms. The authors generated

Table 1
Direct measures to characterize datasets. The symbols not yet defined are: number of attributes (M), number of examples (N), and number of classes (C).

Category	Measure	Description	Complexity
Simple	1. Number of attributes	Depict properties taken from the attribute-value table.	$O(M)$
	2. Number of qualitative attributes		$O(M)$
	3. Number of quantitative attributes		$O(M)$
	4. Number of examples		$O(N)$
	5. Number of classes		$O(C)$
Statistical	1. Average asymmetry of the attributes	Describes how and how much the data distribution departs from the symmetry condition. According to the value assumed by this coefficient, it is possible to classify the distribution as symmetric, moderate asymmetric, or strong asymmetric. In case the distribution is not symmetric, it may be positively or negatively asymmetric.	$O(N.M)$
	2. Average kurtosis of the attributes	Indicates the degree of flatness of a distribution, which is generally considered regarding a Gaussian distribution. In terms of kurtosis, we can categorize the shape of the distribution curve as mesokurtic, platykurtic, or leptokurtic.	$O(N.M)$
	3. Average correlation between attributes	Measures the linear relation degree between random attribute pairs.	$O(N.M)$
	4. Average coefficient of variation of the attributes	Represents the dispersion of data around the average. The lower the value, the more homogeneous the data is.	$O(N.M)$
	5. Balancing of the dataset	Determines the level of balance of the dataset and is defined by the ratio between the number of examples of the classes with the smallest and the largest number of examples. The lower the value, in the interval of zero and one, the greater the imbalance.	$O(N.M)$
	6. Majority class error	Concerns the error in the case of new examples being labeled with the majority class.	$O(N.M)$
Information	1. Class entropy	Reflects the approximate amount of information required to identify the class of an example from the dataset.	$O(N.M)$
	2. Average entropy of the attributes	Estimates the amount of information that a given attribute has to offer in predicting the class.	$O(N.M)$
	3. Average conditional entropy between classes and attributes	Measures the uncertainty degree of the class when the values of a random attribute are unknown.	$O(N.M)$
	4. Average mutual information between classes and attributes	Quantifies the entropy reduction caused by the partition of the examples according to the values of an attribute.	$O(N.M)$
	5. Equivalent number of attributes	Ratio between the entropy of the class and the average mutual information between classes and attributes. Theoretically, it refers to the number of attributes needed to describe the class, considering that they all have the same mutual information with the class.	$O(N.M)$
	6. Signal/noise ratio	Expresses the amount of non-useful information of a dataset and is given by the subtraction of the average mutual information, between classes and attributes, from the average entropy of the attributes; the obtained result is divided by the average mutual information between classes and attributes.	$O(N.M)$

a meta-base using 84 datasets described by 79 input meta-features. These descriptors cover direct characterization measures (simple, statistical, and information-theoretical) and model-based measures. The last group of input meta-features came from the application of a decision tree, *k*NN, and perceptron. Three feature selection algorithms with different strategies and search directions were studied. The experimental protocol adopted five base inductors: BayesNet, C4.5, IB3, Naive Bayes, and PART. Through a wrapper feature selection method, the researchers found an optimal meta-feature subset for each of the classifiers they used to assess feature selection algorithm quality and for the overall case, where it was considered the average performance of the base classifiers.

Goswami et al. (2016) introduced a proposal based on the intra-feature relationship and in a multivariate score measure to group 63 datasets according to their characteristics. The performance of distinct feature selection algorithms on different groups of data was then investigated by simulation experiments to verify the relationship between the dataset proprieties and the feature selection algorithm behavior. The authors also indicated a promising framework regarding the automatic choice of feature selection algorithms.

Parmezan et al. (2017) proposed a new architecture that induces symbolic-based meta-models to recommend feature selection algorithms. The study involved 150 datasets, 21 meta-features belonging to four categories – (i) simple, (ii) statistical, (iii) information-theoretical, and (iv) complexity –, and four feature selection methods based on information, distance, dependence, and consistency measures. Such feature selection algorithms had their performance evaluated in terms of a novel multi-criteria index. According to the researchers, the proposed symbolic meta-models are interpretable, robust to overfitting, and less computationally costly than newly designed approaches in the literature.

Dôres et al. (2017) analyzed the use of metalearning to show when to run feature selection together with 12 classifiers. The experiments considered one ensemble metalearner, 394 datasets, 40 input meta-features (simple, statistical, information-theoretical, landmarkers, and feature selection landmarkers), and one feature selection algorithm. Their results showed that, although there is an advantage in using metalearning, these gains are not yet sufficiently relevant – results of the statistical analysis were not always significant –, which opens avenues for new research to be carried out in the area.

Aduviri et al. (2018) modified an ensemble of gradient boosting decision trees and a neural network model to predict rankings of combinations of feature selection - classification algorithms for gene expression prediction datasets. Besides a collection of statistical and information-theoretical descriptors, the authors used 60 gene expression datasets, four feature selection methods (ReliefF, Fisher Score,

Table 2
Landmarking measures to characterize datasets.

Measure	Description	Complexity
1. Decision node learner	This simple learning algorithm selects the attribute with the highest information gain-ratio. Subsequently, it creates a single node decision tree consisting of the chosen feature as a split node. The idea is to show how informative an attribute is concerning the classification task.	$O(N.\log N)$
2. Randomly chosen node learner	It is a classifier that resides in a single decision node, based on a randomly chosen feature. Such a landmarker, as well as most of the decision node learner variations, endeavors to provide information about the irrelevant features.	$O(N.\log N)$
3. Worst node learner	This learning algorithm selects the attribute with the lowest information gain among all features to generate a single node decision tree.	$O(N.\log N)$
4. Linear discriminant	Resides in a linear combination of attributes that separates the classes as best as possible. Depending on the number of features combined, the derived separation model can be a line, a plane, or a hyperplane.	$O(N.M^2)$
5. Elite 1NN learner	Computes 1-Nearest Neighbor, where the test set is labeled with the class of the closest training example on a subset of all attributes. This elite subset consists of the most informative features if the gain-ratio difference between them is smaller than 0.1. Otherwise, the elite subset is a singleton and the landmarker acts as a decision node learner. The aim is to determine whether the task is a relational one, *i.e.*, if it involves parity-like relationships between the attributes. In relational tasks, no single feature is considerably more informative than the others.	$O(N.M)$
6. Naive Bayes	Constitutes a probabilistic model that uses the training dataset to estimate the probabilities required to use the Bayes' theorem to classify test examples. We can interpret the predictive performance of Naive Bayes as an independence measure between attributes.	$O(N.M)$

Table 3
Model-based measures to characterize datasets. The symbols not yet defined are: number of nodes or vertices (V), and number of edges (E).

Measure	Description	Complexity
1. Tree width	Represents a property of a decision tree learned from a dataset. In particular, it counts the number of lengthways partitions in the tree.	$O(N.M^2+V+E)$
2. Tree height	Counts the number of levels in a decision tree learned from a dataset.	$O(N.M^2+V+E)$
3. Number of nodes	Indicates the number of tree nodes that have children.	$O(N.M^2+V+E)$
4. Number of leaves	Indicates the number of tree nodes that do not have children.	$O(N.M^2+V+E)$
5. Maximum number of nodes in one level	After identifying the number of nodes in each level of a decision tree learned from a dataset, the measure returns the highest value found.	$O(N.M^2+V+E)$
6. Mean number of nodes in one level	Averages the number of nodes in each tree level.	$O(N.M^2+V+E)$
7. Standard deviation of the number of nodes in one level	Calculates the standard deviation of the number of nodes in each tree level.	$O(N.M^2+V+E)$
8. Longest branch	After identifying the length of the branches of a decision tree learned from a dataset, the measure returns the highest value found.	$O(N.M^2+V+E)$
9. Smallest branch	After identifying the length of the tree branches, the measure returns the lowest value found.	$O(N.M^2+V+E)$
10. Mean branch length	Averages the length of tree branches.	$O(N.M^2+V+E)$
11. Standard deviation of the branch length	Calculate the standard deviation of the length of tree branches.	$O(N.M^2+V+E)$
12. Minimum occurrence of attributes	After identifying the frequency of each feature in a tree, the measure returns the lowest value found.	$O(N.M^2+V+E)$
13. Maximum occurrence of attributes	After identifying the frequency of each feature in a decision tree learned from a dataset, the measure returns the highest value found.	$O(N.M^2+V+E)$
14. Mean occurrence of attributes	Averages the frequency of attributes in a tree.	$O(N.M^2+V+E)$
15. Standard deviation of the occurrence of attributes	Calculates the standard deviation of the frequency of attributes in a tree.	$O(N.M^2+V+E)$

Chi2, and Random Forest), and three classification algorithms (SVM, Naive Bayes, and Logistic Regression). The results showed a meaningful gain in performance and stability concerning the conventional *k*NN meta-learner.

As one can see, the previous studies on feature selection algorithm recommendation via metalearning provide positive experimental evidence that encourages the accomplishment of the present study.

3.2. Measures for input meta-feature extraction

Input meta-features are data descriptors that can be used to uniformly describe different datasets, with distinct sizes and different data distributions. Such characterization measures should have two essential proprieties. They need first, to be easy to calculate. Secondly, they should obtain helpful information in finding feature selection algorithms' performance. Input meta-features still need to obtain information relating to dataset configuration and dataset complexity.

Table 4
Complexity measures to characterize datasets.

Measure	Description	Complexity
1. Fractal dimension of the dataset	Reflects the intrinsic dimension of the dataset, so that it can be approximated in a space with lower dimensionality due to the presence of redundancy in the original data.	$O(N.\log N)$
2. Maximum Fisher's discriminant ratio	Expresses the Fisher discrimination rate. A high value means that there is a transformation vector that can separate the examples of distinct classes after their projection in this new feature space.	$O(N.M)$
3. Volume of the overlapping region	Represents the overlap of the limits defined by examples of each class. A low value indicates that attributes can correctly separate examples from distinct classes.	$O(N.M.C)$
4. Dispersion of the dataset	Quantifies the dispersion degree of the data by calculating the ratio between the number of examples and the number of features of the dataset.	$O(M+N)$

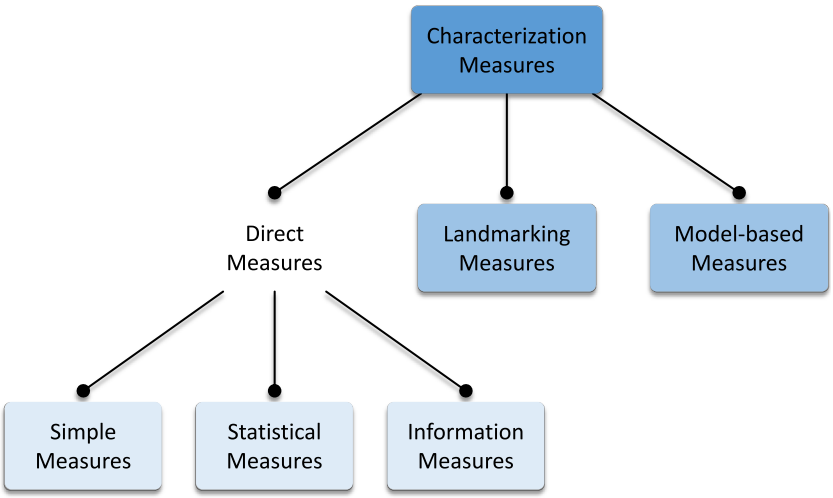

Fig. 5. Hierarchy of classical characterization measures.

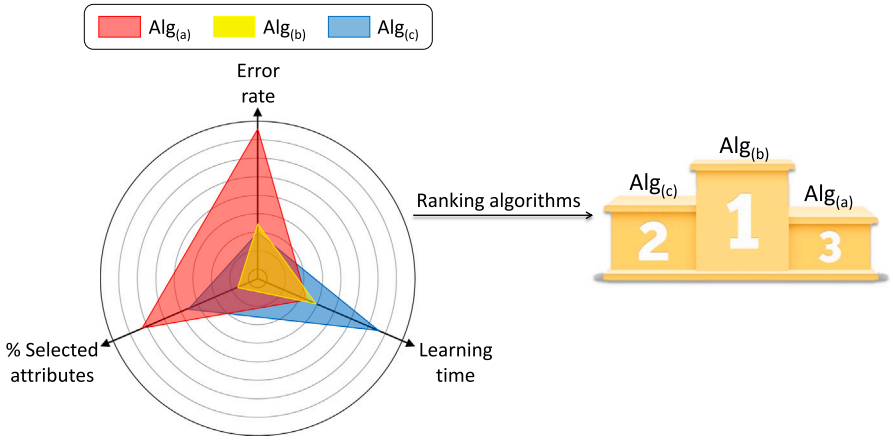

Fig. 6. Multi-criteria performance measure.

The efficiency of the feature selection algorithm recommendation, as well as any other metalearning problem, necessarily depends on the description of the datasets. An unsuitable choice of input meta-features could compromise the entire metalearning process, creating a recommendation model that is inappropriate for the desired purpose.

The research on datasets' characterization for the automatic suggestion of feature selection algorithms focuses on three subareas (Fig. 5) (Reif et al., 2014): (i) direct characterization, which covers simple, statistical, and information-theoretical input meta-features; (ii) characterization by landmarking; and (iii) characterization via models. We describe each of these data descriptor categories as follows. Our supplementary material includes all mathematical formulations of these measures.

Table 1 shows the traditional direct characterization measures (Kalousis, 2002; Parmezan et al., 2017). In this table, measures belonging to the simple category try to describe general characteristics of the datasets, while information theory-based measures seek to characterize the nominal attributes and their relationship with the class attribute. We usually apply statistical measures to verify the order, description, and distribution of numeric data. All these conventional descriptors are computationally inexpensive, and we may use them in several contexts.

The landmarking characterization consists of using simple classification algorithms, called landmarkers, on datasets to get relevant

information regarding the nature of the domain to which they are applied. Landmarking is generally used to determine the proximity of a dataset related to others, through performance (hit rate or accuracy) similarity of landmarkers. Table 2 summarizes the main landmarking descriptors (Pfahringer et al., 2000; Shilbayeh & Vadera, 2014).

The model-based characterization, similar to landmarking, also adopts machine learning methods to represent datasets. Nevertheless, it does not directly take into account performance indexes of the classification algorithm, but the structure of the classifier itself (induced hypothesis or model). Decision tree inducers are commonly used for this task because, from the built tree, it is possible to define a set of input meta-features derived from the number of internal and leaf nodes, as well as the depth of the tree and its height. Table 3 exhibits the input meta-features extracted from decision trees that researchers have been exploring in the related literature (Dôres et al., 2017; Filchenkov & Pendryak, 2015; Peng et al., 2002).

Initially, it may seem that the use of model-based characterization is more worthwhile than other types of characterization, as its employment permits the dataset to be summarized by a data structure that embeds both the performance and complexity of the built classifier and not just the data distribution. The truth is that the obtained representation can, in some cases, mask relevant properties that help explain the performance of the algorithms we want to recommend. Such a problem also impacts the characterization via landmarkers. In this context, the complexity estimate of the dataset is an important issue that, if regarded, can improve the metadata quality.

As far as we know, Parmezan et al. (2017) is the only paper in the metalearning field that applies complexity measures for feature selection algorithm recommendation. Table 4 lists the complexity descriptors introduced in that study. These characterization measures aim to describe how the data are geometrically structured, considering inherent traits such as separability of classes, overlap in attribute space, topology, and density.

3.3. Measures for target meta-feature construction

The feature selection algorithm recommendation is a more challenging problem than the learning algorithm suggestion. We can affirm this because evaluating a feature selection algorithm based on the performance of the model built using the selected subset of features, a typical setting in the literature, adds a new variable to concern: the model itself. In other words, the model used in this scenario gives us an indirect estimate of the feature selection quality. Learning algorithm suggestion does not face the same issue. Hence, in this line of sight, the target meta-feature construction is as important as the input meta-feature engineering.

To obtain a broad estimate of the quality of feature selection algorithms, we must consider multiple evaluation measures, such as the performance of the model induced from the selected subset of features (acc), the runtime of the feature selection (t_α), the percentage of selected features (p), and the learning time for the building of such model (t_β). The model efficiency acc reflects the importance of the chosen features for further processing; the runtime t_α indicates how fast the feature selection algorithm is; the percentage p represents the degree of data compression achieved by the evaluated algorithm; and time t_β shows the training time cost needed to learn a decision model from the selected attributes. Another relevant aspect related to the target meta-feature generation comprises the gain/loss magnitude of the method in terms of the performance of the model induced from the selected subset of features, *e.g.*, the difference in the accuracy of a learning method with and without feature selection.

Many studies from both the data mining and metalearning literature have explored the individual performance indexes described above (Arauzo-Azofra et al., 2011; Bolón-Canedo et al., 2013; Forman, 2003; Molina et al., 2002). Some authors, however, have opted for stability measures to evaluate the quality of feature selection algorithms (Dernoncourt et al., 2014; Somol & Novovicova, 2010). Other researchers went further and proposed multi-criteria performance metrics that combine two or more individual performance indexes in a single one (Parmezan et al., 2017; Wang et al., 2013).

Parmezan et al. (2017) introduced a Multi-Criteria Performance Measure (MCPM) to estimate the quality of feature selection algorithms. The proposal is so robust that it has been used successfully to assess the performance of methods for other tasks, such as dermoscopy image classification (Lee et al., 2018) and time series forecasting (Parmezan et al., 2019). As the first step to apply this measure, it is necessary to choose the performance indexes to be simultaneously considered. Parmezan et al. (2017) suggested the use of three measures: (i) error rate of an ensemble classifier induced from the subsets of attributes selected by feature selection algorithms — such an error ($1 - acc$) is computed in agreement with some validation method, *e.g.* holdout, leave-one-out, or cross-validation; (ii) learning time, in seconds, needed to construct the ensemble model using the subsets of features (t_β); and (iii) the percentage of selected attributes by feature selection algorithms (p).

After configuring the MCPM with these variables, we can obtain a single measure to estimate the predictive performance for each evaluated feature selection method. The multi-criteria measure considers the total area of the irregular triangle, which in turn is defined from the values achieved by the feature selection algorithm in the three variables. The lower the total area, the better the corresponding feature selection method is. To illustrate this idea, Fig. 6 shows the irregular triangles regarding a decision model generated from three feature subsets, each resulting from a feature selection algorithm. In this example, the best feature selection method is $\text{Alg}_{(b)}$ while the worst is $\text{Alg}_{(a)}$.

4. Proposed meta-feature engineering model

Two critical issues inherent to any metalearning system are choosing a suitable data representation and formulating a goal (or goals) of data modeling. With these issues in mind, we designed a novel meta-feature engineering model that introduces unprecedented methods and combines them with state-of-the-art techniques to build more representative meta-examples – input meta-features and target meta-features – for the problem of automatic recommendation of feature selection algorithms.

Fig. 7 outlines our meta-feature engineering model. A total of 161 input meta-features based on eight categories of dataset descriptors were extracted, and a multi-criteria measure was used to rank the feature selection algorithms depending on their performance given by a hybrid ensemble classifier. Within these eight categories of dataset characterization measures, we considered 42 classical ones from the literature and proposed 119 measures in this work.

For didactic purposes, we organized the model into three large processes: (i) input meta-feature extraction — Fig. 7(1); (ii) target meta-feature construction — Fig. 7(2); and (iii) meta-base(s) elaboration — Fig. 7(3). We describe all the steps involved in each of these processes, respectively, in Sections 4.1–4.3.

4.1. Input meta-feature extraction

The input meta-feature generation comprises two steps (Fig. 7(1) — A and B):

Step A — Preprocessing of datasets:

The original dataset repository considers $|D|$ elements, each one corresponding to a dataset from which we extracted input meta-features. We represent each dataset with N examples and M features in the attribute-value format.

One possible way to enrich our data repository is to decompose the multiclass datasets into binary ones. We propose this

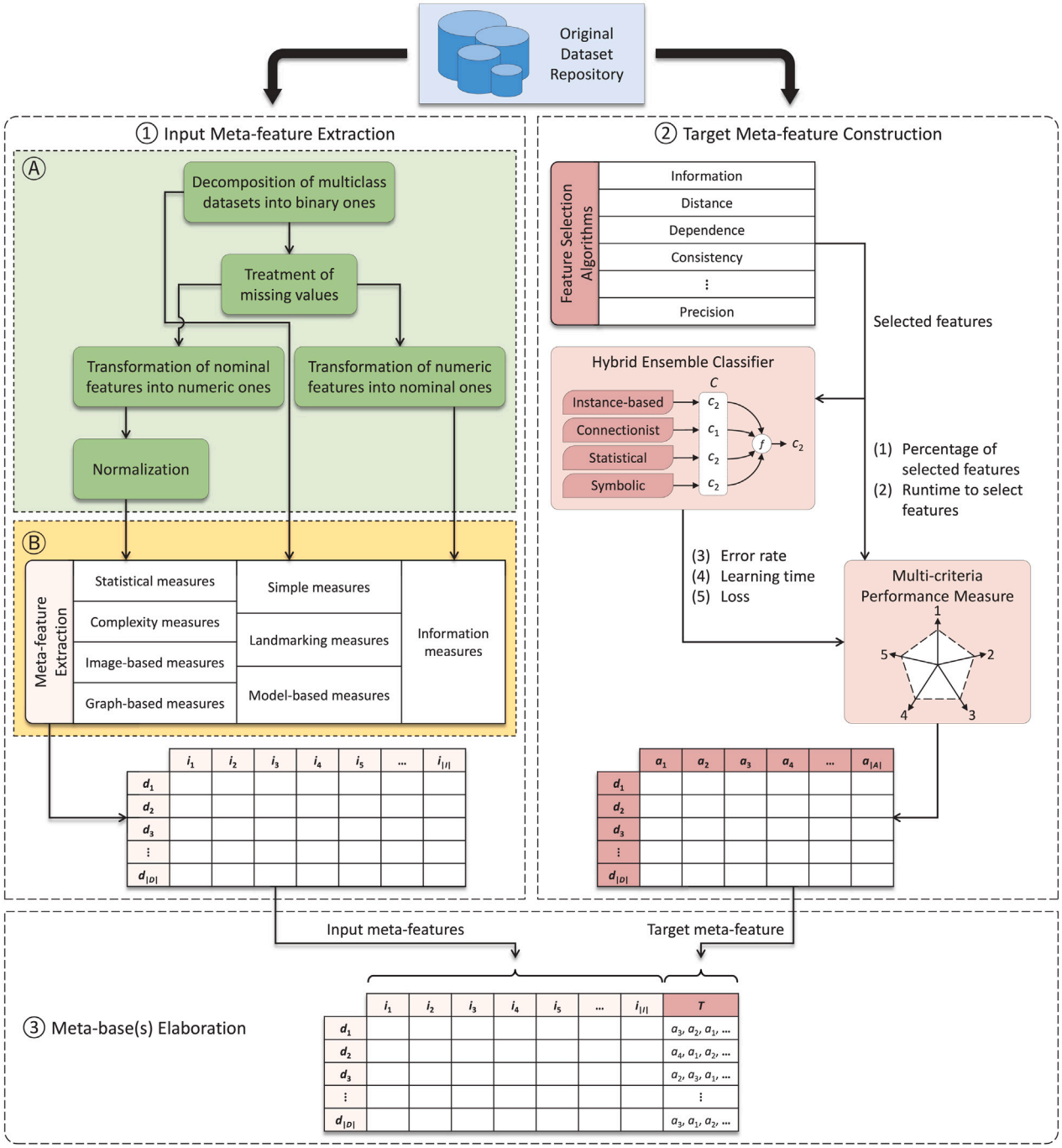

Fig. 7. Proposed meta-feature engineering model.

procedure for the sake of creating more datasets inspired by the multi-label learning approach Binary Relevance (Tsoumakas et al., 2009). Each multiclass dataset with C class values is separated into C pieces, one for each label (class value) in the input dataset, resulting in C independent binary classification datasets. In each binary dataset, we regard the examples associated with the corresponding label as positive and the remnants as negative. However, unlike the multi-label case, each input example is labeled positive in only one binary dataset.

Then, the resultant datasets, which may contain missing values, undergo treatment, as the lack of feature values prevents extracting some characterization measures. Diverse approaches with varying complexities may be applied to treat missing values. They range from the simplest replacement of the missing values by the mode or by the mean, depending on whether the corresponding attribute is nominal or numeric, respectively, up to using machine learning algorithms to predict these missing values (Grzymala-Busse & Hu, 2000).

As some characterization measures function with either qualitative or quantitative types of data, we also propose here the use of supervised feature transformation methods to transform nominal features into numeric ones and vice-versa. The variety of possible supervised algorithms for feature transformation is large (Witten et al., 2011), and the Weka tool provides a good selection of them.

Since numeric features may have different ranges, we must normalize them to a common scale, without distorting differences in

Table 5
New direct measures to characterize datasets.

Category	Measure	Description	Complexity
Simple	1. Number of attributes with missing values	Depict properties taken from the attribute-value table.	$O(N.M)$
	2. Number of examples with missing values		$O(N.M)$
	3. Percentage of missing values		$O(N.M)$
Statistical	1. Minimum asymmetry of the attributes	These descriptors are variations of the asymmetry measure — Table 1.	$O(N.M)$
	2. Maximum asymmetry of the attributes		
	3. Minimum kurtosis of the attributes	These descriptors are variations of the kurtosis measure — Table 1.	$O(N.M)$
	4. Maximum kurtosis of the attributes		
	5. Minimum coefficient of variation of the attributes	These descriptors are variations of the coefficient of variation — Table 1.	$O(N.M)$
	6. Maximum coefficient of variation of the attributes		
Information	1. Minimum entropy of the attributes	These descriptors are variations of the entropy measure — Table 1.	$O(N.M)$
	2. Maximum entropy of the attributes		
	3. Minimum conditional entropy between classes and attributes	These descriptors are variations of the conditional entropy measure — Table 1.	$O(N.M)$
	4. Maximum conditional entropy between classes and attributes		
	5. Minimum mutual information between classes and attributes	These descriptors are variations of the mutual information measure — Table 1.	$O(N.M)$
	6. Maximum mutual information between classes and attributes		

Table 6
New complexity measures to characterize datasets. The symbol not yet defined is: number of interpolated points (L).

Measure	Description	Complexity
1. Intra-class distance	Considers the density of each class in a dataset by averaging the distance between each example and the center of its class across all classes.	$O(N.M)$
2. Inter-class distance	Considers the separability between the classes in a dataset by averaging the distance between each class center and the center of the dataset across all classes.	$O(M.C)$
3. Inconsistent example pairs	Identifies the inconsistency rate of a dataset as the ratio of the number of inconsistent pairs of examples and the total number of pairs of examples in the dataset.	$O(N^2.M)$
4. Attribute class correlation	Highlights datasets with many features showing distinct values for examples of different classes.	$O(N^2.M)$
5. Ratio of intra/extra class nearest neighbor distance	Averages the ratio of the distances between each example and its closest neighbor from the same class (intra-class) and from another class (extra-class). Low values of this measure are indicatives of simpler problems, in which the distance between examples of different classes transcends the distance between examples from the same class. Therefore, this distance-based descriptor reflects how data are distributed within classes and not only how the boundary is between the classes.	$O(N^2.M)$
6. Error rate of the nearest neighbor classifier	Returns the error rate of a 1NN classifier using leave-one-out validation. High values of this measure demonstrate that many examples are near to examples of other classes, *i.e.*, the problem is complex.	$O(N^2.M)$
7. Non-linearity of the nearest neighbor classifier	It first generates a new dataset by interpolating training examples of the same class. Specifically, two examples from the same class are selected randomly and they are linearly interpolated, with the support of random coefficients, producing a new example. Then a 1NN classifier is trained on the original data and has its error rate measured in the new data examples. Higher values of this descriptor are indicative of problems of greater complexity.	$O(M.N.L)$
8. Ratio measure	Estimates the difficulty in analyzing a dataset and finds its important features by taking into account the number of examples, the number of classes, and the number of possible values of a nominal feature.	$O(N.M)$

the ranges of values. We can obtain this normalization by several means, but a simple way is through z-scores (or z-normalization) (Parmezan & Batista, 2015).

Step B — Extraction of characterization measures:

In this step, we extract three groups of dataset characterization measures according to the data source:

- From the decomposed datasets, we extract three types of measures regarding the general characteristics of the datasets: simple (Tables 1 and 5), landmarking (Table 2), and model-based (Table 3);
- From datasets with original and transformed numeric features, four types of measures are extracted: statistical (Tables 1 and 5), complexity (Tables 4 and 6), image-based (Table 7), and graph-based (Table 8);

Table 7
Image-based measures to characterize datasets.

Measure	Description	Complexity
1. Energy	Measures the textural uniformity based on pixel pair repetitions. High energy values indicate that there are more instances of intensity value pairs in the image that neighbor each other at higher frequencies.	$O(N.M)$
2. Contrast	Measures the spatial frequency of an image, *i.e.*, the difference between the highest and the lowest values of a contiguous set of pixels. A low-contrast image exhibits the Gray-Level Co-occurrency Matrix (GLCM) concentration term around the principal diagonal and features low spatial frequencies.	$O(N.M)$
3. Correlation	Measures the linear dependency of gray level between the pixels at the specified positions relative to each other.	$O(N.M)$
4. Variance	This descriptor is a measure of heterogeneity. Variance increases when the gray level values differ from their mean.	$O(N.M)$
5. Homogeneity	Measures image homogeneity as it assumes larger values for smaller gray tone differences in pair elements. In terms of GLCM, homogeneity decreases if contrast increases while energy is kept constant.	$O(N.M)$
6. Sum average	Measures the relationship between occurrences of pairs with lower intensity values and occurrences of pairs with higher intensity values.	$O(N.M)$
7. Sum variance	This descriptor is a measure of heterogeneity that assigns higher weights on neighboring intensity level pairs which deviate more from the mean.	$O(N.M)$
8. Sum entropy	It is a sum of neighborhood intensity value differences.	$O(N.M)$
9. Entropy	It measures the complexity or disorder of an image. A high entropy value results from an image that is not texturally uniform.	$O(N.M)$
10. Difference variance	This descriptor is a measure of heterogeneity that assigns higher weights on differing intensity level pairs which deviate more from the mean.	$O(N.M)$
11. Difference entropy	Measures the disorder associated with the gray level difference distribution of the image.	$O(N.M)$
12. Informational measure of correlation I 13. Informational measure of correlation II 14. Maximal correlation coefficient	Informational measures of correlation and the maximal correlation coefficient present some useful properties that are not available from the simple GLCM correlation.	$O(N.M)$

- From datasets with original and transformed nominal features, we extract information measures (Tables 1 and 5).

After extracting all 161 descriptors, the first part of the metabase containing $|I|$ input meta-features is ready to be joined with the target meta-features built by the process introduced in Section 4.2 — Fig. 7(2).

We organize the input meta-features implemented in our model into eight categories. Fig. 8 represents hierarchically these categories, where five of them – simple, statistical, information, landmarking, and model-based measures – belong to the classical automated machine learning field (Fig. 5). Here we are proposing the use of the remaining categories to characterize datasets: complexity, image-based, and graph-based measures.

As mentioned, besides selecting 42 descriptors from the metalearning literature (Tables 1, Table 2, Table 3, and Table 4), we also adopted 119 new data characterization measures never before studied in the context of automatic recommendation of feature selection algorithms (Tables 5, Table 6, Table 7, and Table 8). Although this paper is self-contained, further details of all these input meta-features are available in our supplementary material.

Most descriptor categories in Fig. 8 deal with a flat attribute-value table representation of a dataset. We can apply simple, landmarking, and model-based measures to multivariate datasets, *i.e.*, datasets composed by numeric and nominal features, without the need for data transformation. Statistical and information descriptors follow a rule determined by their respective categories, as indicated in Fig. 7(1)-B. The former category works exclusively with numeric features, while the measures of the latter one operate solely on nominal features. An exception occurs for the complexity category. Two descriptors from this category (Attribute class correlation and Ratio measure) are applied on nominal data, instead of numeric data, as the other complexity characterization measures do.

Besides the attribute-value format, other ways to represent a dataset is by using images and graphs, as addressed in detail in the rest of this section. In the case of images, each horizontal line's set of pixels describes an example in the dataset; each pixel corresponds to an attribute value. Such representation reveals the global information associated with the class attribute. In the case of graphs, each vertex is equivalent to an example in the dataset; edges represent the relationships between examples. This representation is useful to describe local information concerning pairs of objects, such as similarities or distances between them.

In this paper, we investigate the use of characterization measures based on images and graphs. To do so, we developed two methods capable of converting numeric datasets into grayscale images and undirected graphs, respectively. We explain these methods as follows.

Fig. 9 illustrates the three steps required to transform numeric data into images using the well-known Iris dataset (Fisher, 1936). We begin Step 1 by determining the centroids of the examples for each class in the dataset; the centroid is the average of all the examples (multidimensional vectors) labeled with a specific class. After that, we calculate the distances between the examples and their respective centroids. The distances are then used as a reference to sort the examples of each class, from the nearest to the most distant.

In Step 2, we transform the numeric data into the grayscale color domain via min–max normalization. Although there are several methods for normalization (Han et al., 2011), min–max is one of the simplest that preserves the integrity of data through scaling. Eq. (1) defines the min–max normalization, where $\hat{x}$ is the normalized pixel, x denotes the pixel of interest, x_{max} and x_{min} are, in this order, the maximum and minimum values of attribute X ($x \in X$), and α and β correspond to the minimum and maximum values of the color space.

$$\hat{x} = \alpha + \frac{(x - x_{min})(\beta - \alpha)}{x_{max} - x_{min}} \tag{1}$$

As the color space is 8 bit, we scale the numeric values that make up the dataset to the interval $[0, 255]$. Typically, zero is taken to be black,

Table 8
Graph-based measures to characterize datasets. The symbol not yet defined is: average degree (G).

Measure	Description	Complexity
1. Number of edges	Property associated with the size of the graph.	$O(E)$
2. Edge density	Ratio between the number of edges and the number of possible edges.	$O(V+E)$
3. Minimum degree 4. Maximum degree 5. Average degree 6. Median degree 7. Mode degree	Degree is a measure applied for each vertex of the graph. The degree of a vertex is the total number of its adjacent edges.	$O(V.G)$
8. Minimum strength (weighted graph) 9. Maximum strength (weighted graph) 10. Average strength (weighted graph) 11. Median strength (weighted graph)	Strength (or degree for the unweighted version) is a measure computed for each vertex of the graph. The strength of a vertex is the total sum of the weights of its adjacent edges.	$O(V+E)$
12. Diameter 13. Diameter (weighted graph)	The diameter is the maximum geodesic distance between any pair of vertices in the graph. In other words, it is the length of the longest geodesic between any pair of vertices.	$O(V.E)$
14. Minimum closeness centrality 15. Maximum closeness centrality 16. Average closeness centrality 17. Median closeness centrality	Closeness centrality is a measure computed for each vertex of the graph. It calculates how many steps are needed to access every other vertex from a given vertex.	$O(V.E)$
18. Minimum betweenness centrality 19. Maximum betweenness centrality 20. Average betweenness centrality 21. Median betweenness centrality	The betweenness centrality for each vertex is the number of geodesics (shortest paths) that pass through the vertex.	$O(V.E)$
22. Correlation degree centrality	Corresponds to the correlation between the degree of the graph and the subgraph centrality of the vertices. The subgraph centrality of a vertex computes the number of subgraphs a vertex participates in, weighting them according to their size.	$O(V^3)$
23. Assortativity	Refers to the Pearson correlation coefficient of degree between pairs of connected vertices. Positive assortative values reveal a correlation between vertices of similar degree, while negative values expose relationships between vertices of different degree.	$O(E)$
24. Minimum constraint 25. Maximum constraint 26. Average constraint 27. Median constraint	The constraint measure calculates Burt's constraint for each vertex of the graph. The main idea is to determine the extent to which a vertex $\mathbb{V}$ is invested in those vertices that are themselves invested in the neighbors of $\mathbb{V}$.	$O(V^3)$
28. Minimum coreness 29. Maximum coreness 30. Average coreness 31. Median coreness 32. Mode coreness	The k-core of a graph is the largest subgraph where each vertex has at least degree k. The coreness of a vertex is k if it pertains to the k-core but not to the $(k+1)$-core.	$O(E)$
33. Minimum diversity (weighted graph) 34. Maximum diversity (weighted graph) 35. Average diversity (weighted graph) 36. Median diversity (weighted graph)	Diversity is a measure computed for each vertex of the graph. We defined diversity as the Shannon entropy of the weights of its incident edges.	$O(V+E)$
37. Minimum eccentricity 38. Maximum eccentricity 39. Average eccentricity 40. Median eccentricity 41. Mode eccentricity	Eccentricity is a measure applied for each vertex of the graph. The eccentricity of a vertex $\mathbb{V}$ is the maximum distance from $\mathbb{V}$ to any vertex.	$O(V^2+V.E)$
42. Number of triangles	Given a graph, it counts how many triangles each vertex is part of.	$O(V^2)$
43. Minimum clustering coefficient 44. Maximum clustering coefficient 45. Average clustering coefficient 46. Median clustering coefficient 47. Minimum clustering coefficient (weighted graph) 48. Maximum clustering coefficient (weighted graph) 49. Average clustering coefficient (weighted graph) 50. Median clustering coefficient (weighted graph)	The clustering coefficient of a vertex is the ratio of the triangles connected to the vertex and the triples centered on the vertex. It estimates how well connected the neighborhood of the vertex is. The weighted clustering coefficient, in turn, is a measure that combines topological information with the weight distribution of the graph. Here we use the definition by A. Barrat, which considers only weights of edges adjacent to the vertex of interest but not the weights of edges between neighbors of the investigated vertex.	$O(V.G^2)$
51. Minimum eigenvector centrality 52. Maximum eigenvector centrality 53. Average eigenvector centrality 54. Eigenvalue associate to the calculated eigenvector 55. Minimum eigenvector centrality (weighted graph) 56. Maximum eigenvector centrality (weighted graph) 57. Average eigenvector centrality (weighted graph) 58. Eigenvalue associate to the calculated eigenvector (weighted graph)	The eigenvector centrality assigns relative scores to all vertices in the graph based on the concept that connections to high-scoring vertices contribute more to the score of the vertex in question than connections to low-scoring vertices. In general, vertices with high eigenvector centralities are connected to many other vertices, which, in turn, are connected to many others.	$O(V)$

(*continued on next page*)

Table 8 (*continued*).

Measure	Description	Complexity
59. Minimum authority score 60. Maximum authority score 61. Average authority score 62. Eigenvalue associate to the calculated eigenvector regarding the authority score 63. Minimum authority score (weighted graph) 64. Maximum authority score (weighted graph) 65. Average authority score (weighted graph) 66. Eigenvalue associate to the calculated eigenvector regarding the authority score (weighted graph)	The authority scores of the vertices correspond to the principal eigenvector of $\mathbb{A}^T \times \mathbb{A}$, in which $\mathbb{A}$ is the adjacency matrix of the graph and $\mathbb{A}^T$ is its transpose. A vertex has a high authority score if many vertices with high hub scores point to it. Hence, good hubs are those which point to many good authorities and good authorities are those pointed to by many good hubs. For undirected graphs, the adjacency matrix $\mathbb{A}$ is symmetric and the authority scores are the same as hub scores. The measure used to calculate the authority scores in weighted graphs interprets the vertex edge weights as connection strengths. Thus, we can argue that a vertex is important because it links to other important vertices through edges with high weights.	$O(V)$
67. Minimum page rank 68. Maximum page rank 69. Average page rank 70. Median page rank 71. Minimum page rank (weighted graph) 72. Maximum page rank (weighted graph) 73. Average page rank (weighted graph) 74. Median page rank (weighted graph)	Google's PageRank is a variant of eigenvector centrality that assigns a numeric weight to each vertex in the graph, intending to estimate its relative importance within the set of vertices. We can compute it by either iteratively distributing one vertex's rank – originally based on degree – over its neighbors or by randomly traversing the graph and counting the frequency of hitting each vertex during these walks. The links are also weighted depending on the relative score of its originating node. The result is that vertices with several incoming links are influential, and nodes to which they are linked share some of that influence.	$O(E)$
75. Minimum local scan 76. Maximum local scan 77. Average local scan 78. Median local scan 79. Mode local scan	The scan statistic is a summary of the locality statistics that is computed from the local neighborhood of each vertex.	$O(V^2.G + V.E)$
80. Number of groups	Refers to the number of clusters (groups of vertices) found. We can perform clustering by calculating the maximal connected components of the graph.	$O(V + E)$
81. Modularity	Corresponds to the modularity score, *i.e.*, how modular is a given division of a graph into subgraphs (communities). A simple way to detect communities is through the use of the fast greedy modularity optimization algorithm.	$O(E)$
82. Number of communities	Indicates the number of dense subgraphs (communities) in a graph. We can find community structures employing the fast greedy modularity optimization algorithm.	$O(E)$

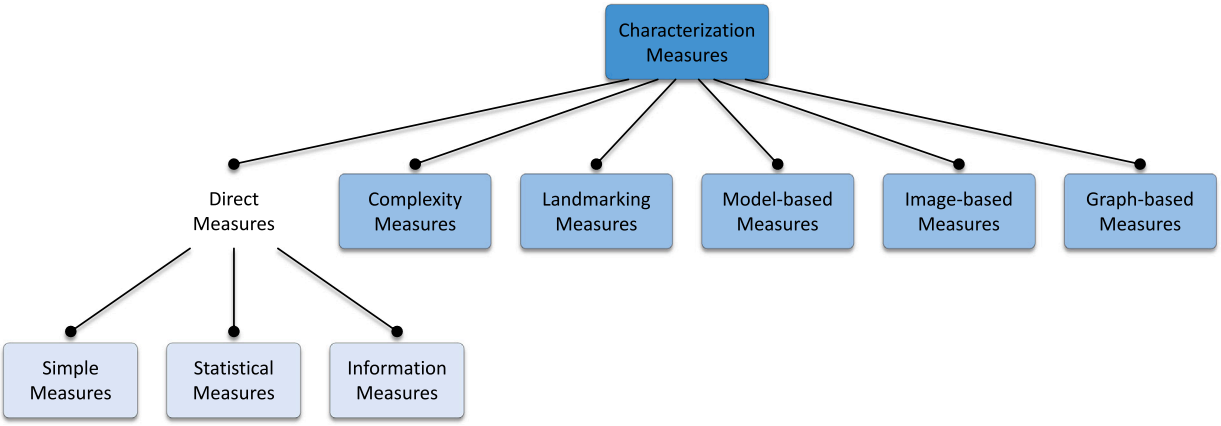

Fig. 8. Extended hierarchy of characterization measures.

and 255 is considered white. Values in between constitute the different shades of gray. Therefore, setting $\alpha = 0$ and $\beta = 255$ defines Eq. (2).

$$\hat{x} = \frac{(x - x_{min})255}{x_{max} - x_{min}} \tag{2}$$

The application of Eq. (2) on the numeric data sample produces a dataset normalized into the grayscale color space. This grayscale image – generated using the entire dataset – can now be provided as input to Step 3, which is responsible not only for dividing it according to the class attribute but also for creating Gray-Level Co-occurrence Matrices (GLCM) from the derived images. GLCM displays how often each gray level occurs at a pixel situated at a fixed geometric position relative to each other pixel, as a function of the grayscale. From a GLCM, we can extract image-based descriptors, such as those of texture proposed in Haralick et al. (1973) (Table 7).

The second data conversion method developed in this work enables the conversion of numeric datasets into graphs. Fig. 10 exemplifies this process in three steps using the Iris dataset. In Step 1, we ignore the class attribute and construct a distance matrix from the numeric dataset. The distance matrix is a two-dimensional array (square matrix) containing the distances, taken pairwise, between the examples of the dataset under consideration.

We feed Step 2 with the distance matrix computed in Step 1. The purpose here is to create a graph taking into account only the matrix of calculated distances. The complex network literature considers some techniques that can be used to build such a graph, also called network,

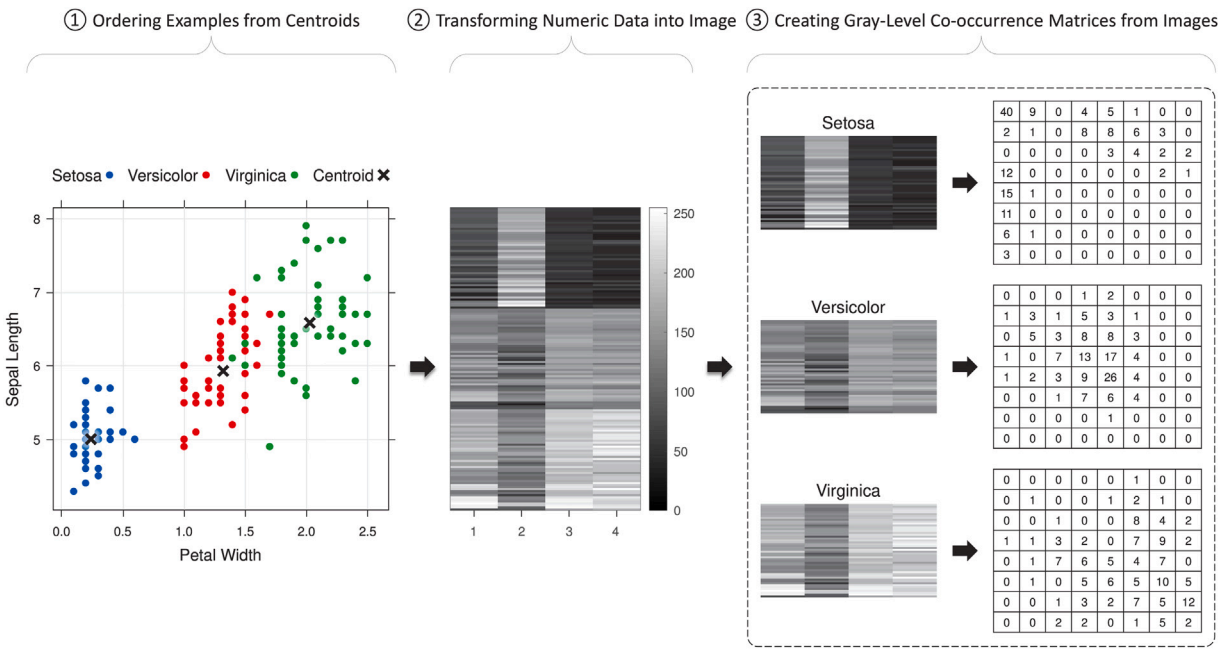

Fig. 9. Transforming the Iris dataset into an image.

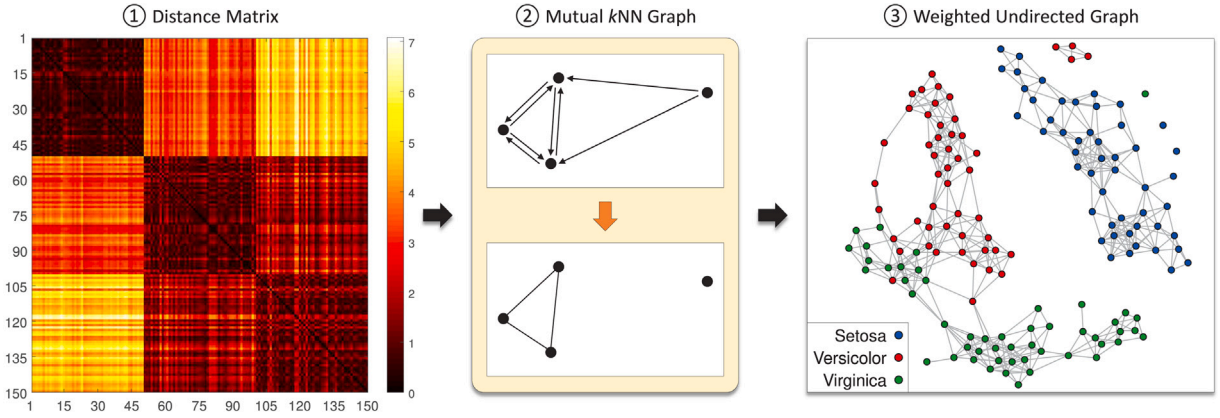

Fig. 10. Mutual *k*NN graph on the Iris dataset.

from a matrix representation of a dataset. One approach is a fully connected graph, which covers an edge between all pairs of vertices (examples). These edges are weighted so that similar vertices have a large edge weight between them. Another way is to make sparse networks via *k*NN graph or ϵNN graph. In *k*NN graphs, vertices i and j are linked by an edge if i is in the k-nearest-neighborhood of j. Parameter k controls the density of the graph. For ϵNN graphs, vertices i and j are connected by an edge, if the distance $d(i, j) < \epsilon$. Parameter ϵ controls the neighborhood radius.

In the present study, we propose using the Mutual *k*NN graph to represent numeric data topologically in the context of metalearning. The Mutual *k*NN graph is a subgraph of a *k*NN graph in which there is an edge between vertices i and j if i is one of the k-nearest neighbors of j and j is one of the k-nearest neighbors of i. In Step 3 of Fig. 10, we can see the undirected graph with weighted edges generated from the Iris dataset using the Mutual *k*NN graph. In this figure, we do not display the edge weights for the sake of simplicity. We must also emphasize that it is possible to discard the weights of the generated graph to explore an unweighted graph as input for some characterization measures.

A complex network is a graph with non-trivial topological attributes, *i.e.*, features that often occur in real networks. A topic of interest associated with these networks is their characterization by means of some graph-based measures (Costa et al., 2007). In this paper, we apply the leading measures of complex networks (Table 8) to characterize datasets coming from the most diverse fields of application. Our idea is to show that structural measures of datasets, which are obtained by representing the dataset as a graph, can be used to enhance the representation quality of meta-examples throughout the process of choosing feature selection algorithms using metalearning.

4.2. Target meta-feature construction

We start the process depicted in Fig. 7(2) by running feature selection algorithms, from different approaches and using various importance measures to decide which features are to be considered significant or not. We then obtain a set of selected features for each element, *i.e.*, dataset in the original dataset repository. These selected features and related information will feed two other parts of the current process:

- We propose using a hybrid ensemble classifier to evaluate feature selection algorithms considering the bias of different machine learning algorithms. The idea behind this model is to combine conceptually distinct base classifiers and employ a majority vote

Table 9
Feature selection algorithms and their proprieties.

		CBF	CFS	InfoGain	ReliefF	WSE
Evaluation strategy	Individual evaluation			✓	✓	
	Subset evaluation	✓	✓			✓
Importance measure category	Consistency measure	✓				
	Dependence measure		✓			
	Distance measure				✓	
	Information measure			✓		
	Precision measure					✓
Interaction with the learning algorithm	Filter approach	✓	✓	✓	✓	
	Wrapper approach					✓

to predict the class labels. We recommend classification methods of different learning paradigms, such as instance-based, connectionist, statistical and symbolic (Han et al., 2011), as base inductors. The hybrid ensemble model receives the selected features and will provide the following information to the MCPM: (3) error rate, (4) learning time, and (5) loss (or gain complement). We formulate the $gain$ as being the ratio between the error rate of the hybrid ensemble classifier from the selected features and the Majority Class Error (MCE) — so-called naive model. Gain values less than 1 indicate that hybrid ensemble classifiers outperformed naive models. Gain values greater than 1 indicate the opposite. We adopted the gain complement, given by $|1 - gain|$, to minimize the MCPM;

- The MCPM receives the (1) percentage of selected features and (2) runtime to select features, and combines them with (3), (4), and (5) to estimate the quality of feature selection algorithms as a single measure.

As a result, for each repository dataset, we have a ranking of the evaluated algorithms concerning their performance computed from the merge of five individual evaluation indexes. We incorporate this ranking as a descent ordered list of the best feature selection algorithms acting as the target meta-feature T.

4.3. Meta-base(s) elaboration

This process, depicted in Fig. 7(3), associates the input meta-features I with the nominal values of the target meta-feature T to compose meta-examples. Recall that the meta-features I correspond to the characterization measures. Also, each T value consists of a list containing a set of labels with length $|A|$, where a label a_i represents a feature selection algorithm, and its position i reflects its performance in decreasing order according to the MCPM.

We call the set of $|D|$ meta-examples "meta-base". From a meta-base, we can apply supervised machine learning algorithms to induce meta-models (models to recommend feature selection algorithms).

There are different ways of conceiving a meta-model. In this paper, we defend the idea of treating the problem of feature selection algorithm recommendation as a chain of binary or multiclass classifiers. Therefore, when three or more labels constitute the target meta-feature and the feature selection algorithms associated with these labels are different, we suggest the decomposition of such nominal feature intending to generate only meta-bases with binary values of target meta-feature. Otherwise, when there are two or more labels and they refer to the same feature selection algorithm, but each with a distinct parameterization, we recommend creating a multiclass meta-base, *i.e.*, a meta-base whose target meta-feature includes all the closely correlated labels. To clarify this idea, let us consider the example of Fig. 11.

Fig. 11 contains a meta-base MB with a target meta-feature T filled by different sequences of a_1, a_2, and a_3 – ordered according to the MCPM –, where a_1 and a_3 are filter feature selection algorithms that evaluate attributes based on information and consistency measures, respectively; and a_2 is a wrapper feature selection algorithm that uses a precision measure as a criterion to select subsets of attributes. We can decompose the problem in MB using background knowledge into two meta-bases (MBX and MBY) if we are interested in recommending the best feature selection algorithm from the three available. In MBX, the target meta-feature took the labels "Filter" and "Wrapper", which we elected following the proprieties of the feature selection algorithm that occupies the first position in the ranked list of candidate algorithms. Analogously in MBY, the target meta-feature included the labels "Consistency" and "Information".

In a nutshell, MBX describes the approach that the feature selection follows. If wrapper, the precision-based algorithm is the most prominent; if filter, this decision is made using MBY, which covers cases where the best feature selection algorithm is either based on consistency or information.

5. Experimental evaluation

We applied the proposed meta-feature engineering model to recommend feature selection algorithms based on measures of consistency, dependence, information, distance, and precision. In this section, we describe the adopted algorithms, the experimental setup, including the evaluation mode of the implemented metalearning framework, as well as the considered datasets.

5.1. Feature selection algorithms and metalearners

This subsection briefly describes five feature selection algorithms and five machine learning classification methods used as metalearners in the experimental evaluation. Let N and M be the number of examples and features, respectively. Let T_2 be the time required to calculate information gain and T_r be the number of decision trees in an ensemble.

Table 9 summarizes some relevant properties of the five traditional feature selection algorithms used in this work (Han et al., 2011; Liu & Motoda, 2007; Witten et al., 2011).

Consistency-Based Filter (CBF) searches for a small feature subset highly consistent with the class with complexity $O(N.M^2)$. Correlation-based Feature Selection (CFS) aims to find, with complexity $O(N.M^2)$, a subset of features highly correlated to the class with small feature-feature correlation. InfoGain estimates the entropy-based information gain of each feature with complexity $O(M.T_2)$. ReliefF rewards features that separate well examples from different classes within a sample of examples with complexity $O(N^2.M)$. In this study, we selected the t features best-ranked by InfoGain and ReliefF to specify the corresponding subsets of features for further evaluation. Wrapper Subset Evaluator (WSE) evaluates each candidate feature subset by estimating the performance of a simple Naive Bayes classifier built from the data described by this subset with complexity $O(N.M)$.

This paper considered as meta-learners five well-known learning algorithms (Chen & Guestrin, 2016; Han et al., 2011; Witten et al., 2011). Symbolic C4.5 uses the divide and conquer strategy to induce, with complexity $O(N.M^2)$, a decision tree in which each node tests a data feature. Random Forest (RF) bootstraps training examples and randomly selects features to build each decision tree in an ensemble, which in turn yields a prediction with complexity $O(T_r.N.M^2)$ by combining

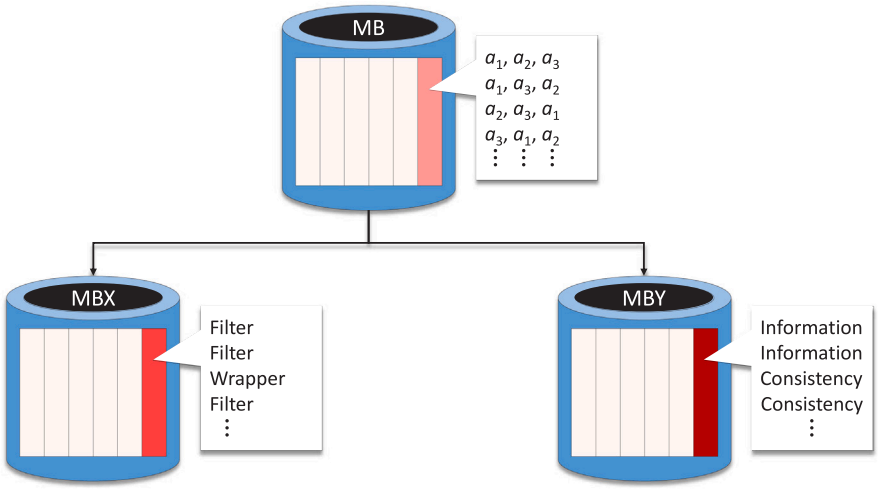

Fig. 11. Target meta-feature decomposition.

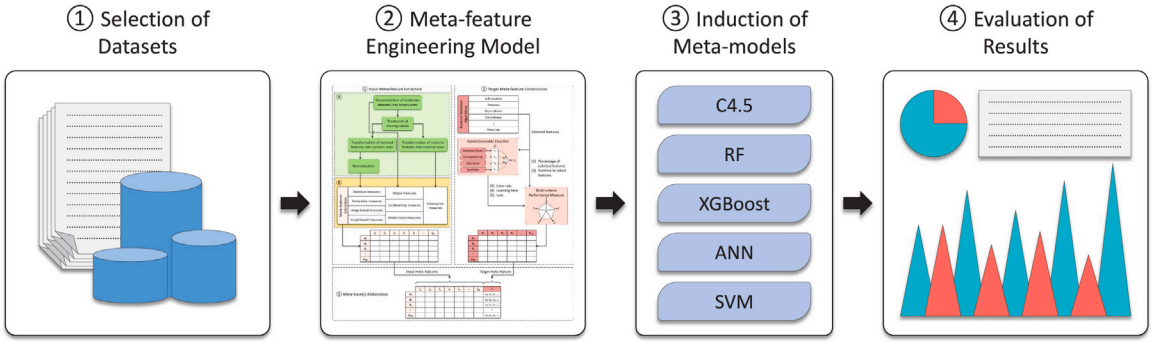

Fig. 12. Experimental setup.

the outputs of the group of trees. XGBoost illustrates another ensemble-based algorithm, but different from RF, XGBoost builds with complexity $O(T_r.N.M^2.\log N)$ a series of decision trees, in which the classification of examples incorrectly predicted by a tree is prioritized by the next tree. Connectionist Multilayer Perceptron combines layers of simple units in an ANN with complexity $O(N.M^2)$, backpropagating prediction errors to adjust the weights of its unit-unit connections. Non-parametric SVM aims to minimize the structural risk by mapping training data into a higher-dimensional space and estimating hyperplanes to separate examples from different classes with complexity $O(N^3)$.

5.2. Experimental setup

We evaluated the proposed meta-feature engineering model following the four steps outlined in Fig. 12.

In Step 1, we have selected 213 datasets from different domains. They are publicly available at the University of California at Irvine (UCI),[2] Uniwersytet Warszawski (UW),[3] Universidad Pablo de Olavo,[4] OpenML (OM),[5] and Other websites (Other). Table 10 summarizes some characteristics of these datasets. In this table, for each dataset, the following is described: number of examples (#E), total number of features (#F), as well as number of qualitative (#FQL) and quantitative features (#FQT), number of classes (#C), Majority Class Error in percentage (MCE), existence of missing values (?), the log of the ratio #F/#E (R), existence of qualitative and quantitative features in the same dataset (B), Domain (D), and Source (S).

In Step 2, we applied the proposed meta-feature engineering model to the 213 datasets without identifiers. Here we follow the same steps illustrated in Fig. 7 with parameters configured as follows: missing values were replaced by the mode or by the mean when the corresponding attribute was nominal or numeric, respectively; Minimum Description Length (Fayyad & Irani, 1993) transformed all numeric features into nominal ones, yielding qualitative datasets; NominalToBinary (Breiman et al., 1984) and z-normalization methods converted nominal features from the processed datasets into numeric ones, generating quantitative datasets; 5-fold cross-validation evaluated landmarkers; we extracted model-based measures from C4.5 classifiers, image-based descriptors from GLCM based on horizontal proximity of the pixels (offset = [0 1]), and graph-based measures from complex networks generated using a Mutual 10NN. Mutual 10NN creates an undirected graph, where each edge is weighted by the normalized distance between the connected vertices; all five feature selection algorithms selected attributes using the full training set (original dataset). We calibrated CBF and CFS with the Best First (BF) and Greedy Stepwise (GS) search methods, both with forward-search. As for InfoGain and ReliefF, we adopted four threshold settings for creating features subsets. Such thresholds allowed the subsets to be composed of τ = 10%, 20%, 40%, and 80% of the best-ranked features. For WSE, we considered Naive Bayes as a base learning algorithm as well as the BF and GS search techniques with forward-search (Witten et al., 2011); hybrid ensemble classifier used the majority voting rule with the following four base machine learning algorithms parameterized in the same way as in Parmezan et al. (2017): Naive Bayes, C4.5, 5NN, and ANN. We evaluated the ensemble model using 10-fold stratified cross-validation.

[2] http://archive.ics.uci.edu/ml.
[3] http://tunedit.org/repo.
[4] http://www.upo.es/eps/bigs/datasets.html.
[5] https://www.openml.org.

Table 10
Summary of characteristics of the datasets.

ID	Dataset	#E	#F	#FQL	#FQT	#C	MCE	?	R	B	D	S
1	acute_inflammations	120	6	5	1	2	49.17		−1.30	✓	Life	UCI
2	ada_agnostic	4562	48	0	48	2	24.81		−1.98		Business	UW
3	ada_prior	4562	14	8	6	2	24.81	✓	−2.51	✓	Business	UW
4	adult_census	32561	14	8	6	2	24.08	✓	−3.37	✓	Social	UCI
5	aids	50	4	2	2	2	50.00		−1.10	✓	Life	OM
6	amazon_commerce_reviews	1500	10000	0	10000	50	98.00		0.82		Physics	UCI
7	analcatdata_authorship	841	70	0	70	4	62.31		−1.08		N/A	OM
8	analcatdata_dmft	797	4	4	0	6	80.55		−2.30		N/A	OM
9	analcatdata_lawsuit	264	4	1	3	2	7.20		−1.82	✓	N/A	OM
10	anneal	898	38	32	6	5	23.83		−1.37	✓	N/A	UW
11	ap_breast_colon	630	10936	0	10936	2	45.40		1.24		Life	OM
12	ap_breast_kidney	604	10936	0	10936	2	43.05		1.26		Life	OM
13	ap_breast_ovary	542	10936	0	10936	2	36.53		1.30		Life	OM
14	ap_colon_kidney	546	10936	0	10936	2	47.62		1.30		Life	OM
15	ap_endometrium_prostate	130	10936	0	10936	2	46.92		1.92		Life	OM
16	arcene	200	10000	0	10000	2	44.00		1.70		Life	UCI
17	arrhythmia	452	279	73	206	13	45.80	✓	−0.21	✓	Life	UCI
18	artificial_characters	10218	7	0	7	10	86.14		−3.16		Computer	UCI
19	audiology	226	69	69	0	24	74.78	✓	−0.52		Life	UCI
20	australian	690	14	8	6	2	44.49		−1.69	✓	Financial	UCI
21	autos	205	25	10	15	6	67.32	✓	−0.91	✓	N/A	UCI
22	backache	180	31	26	5	2	13.89		−0.76	✓	N/A	OM
23	balance	17	3	0	3	2	47.06		−0.75		N/A	UW
24	balance_scale	625	4	0	4	3	53.92		−2.19		Social	UCI
25	balloons	76	4	4	0	2	46.05		−1.28		Social	UCI
26	banana	5300	2	0	2	2	44.83		−3.42		N/A	OM
27	bank	600	10	8	2	2	45.67		−1.78	✓	N/A	Other
28	biomed	209	8	1	7	2	35.89	✓	−1.42	✓	N/A	OM
29	bioresponse	3751	1776	0	1776	2	45.77		−0.32		Life	OM
30	blood_transfusion	748	4	0	4	2	23.80		−2.27		Business	UCI
31	bolts	40	7	0	7	2	35.00		−0.76		N/A	UW
32	brazil_tourism	412	8	4	4	7	22.82	✓	−1.71	✓	N/A	OM
33	breast_cancer	286	9	9	0	2	29.72	✓	−1.50		Life	UCI
34	breast_wisconsin	699	9	0	9	2	34.48	✓	−1.89		Life	UCI
35	bridges_version1	105	11	8	3	6	58.10	✓	−0.98	✓	N/A	UCI
36	bridges_version2	105	11	11	0	6	58.10	✓	−0.98		N/A	UCI
37	bupa	345	6	0	6	2	42.03		−1.76		Life	UCI
38	calories	40	2	0	2	3	50.00	✓	−1.30		N/A	UW
39	car	1728	6	6	0	4	29.98		−2.46		N/A	UCI
40	cardiotocography	2126	35	0	35	10	72.77		−1.78		Life	UCI
41	cars_with_names	406	8	2	6	3	37.44	✓	−1.71	✓	N/A	UCI
42	click_prediction_small	39948	11	0	11	2	16.84		−3.56		Computer	OM
43	climate_model_simulation_crashes	540	20	0	20	2	8.52		−1.43		Physics	UCI
44	cloud	108	7	1	6	2	29.63		−1.19	✓	N/A	UW
45	cnae9	1080	856	0	856	9	88.89		−0.10		Business	UCI
46	colic	368	22	15	7	2	36.96	✓	−1.22	✓	Life	UCI
47	colic_original	368	27	20	7	2	33.70	✓	−1.13	✓	Life	UCI
48	collins	500	22	2	20	15	84.00		−1.36	✓	N/A	OM
49	colon	62	2000	0	2000	2	35.48		1.51		Life	UW
50	companies	79	6	0	6	9	78.48		−1.12		N/A	Other
51	contact_lenses	24	4	4	0	3	37.50		−0.78		N/A	UCI
52	contraceptive_method	1473	9	7	2	3	57.30		−2.21	✓	Life	UCI
53	costa_madre1	296	37	0	37	2	12.84		−0.90		N/A	OM
54	credit_a	690	15	9	6	2	44.49	✓	−1.66	✓	Financial	UCI
55	credit_g	1000	20	13	7	2	30.00		−1.70	✓	Financial	UCI
56	cylinder_bands	540	39	21	18	2	42.22	✓	−1.14	✓	Physics	UCI
57	dermatology	366	34	33	1	6	69.40	✓	−1.03	✓	Life	UCI
58	desharnais	81	12	0	12	3	43.21		−0.83		N/A	OM
59	diagnosis	120	6	5	1	4	66.67		−1.30	✓	Life	UCI
60	dresses_sales	500	12	11	1	2	42.00	✓	−1.62	✓	Computer	UCI
61	ecml_2004	90	27679	0	27679	43	94.44		2.49		N/A	UW
62	ecoli	336	7	0	7	8	57.44		−1.68		Life	UCI
63	eggs	48	3	1	2	2	50.00		−1.20	✓	N/A	UW
64	electricity_normalized	45312	8	1	7	2	42.45		−3.75	✓	N/A	OM
65	embryonal	60	7129	0	7129	2	35.00		2.07		N/A	OM
66	eucalyptus	736	19	5	14	5	70.92	✓	−1.59	✓	Agriculture	UW
67	eye_movements	10936	27	3	24	3	61.03		−2.61	✓	N/A	UW
68	fertility	100	9	6	3	2	12.00		−1.05	✓	Life	UCI
69	first_order_theorem	6118	51	0	51	6	58.25		−2.08		Computer	UCI
70	flags	194	28	26	2	8	64.43		−0.84	✓	N/A	UCI
71	gamma_telescope	19020	10	0	10	2	35.16		−3.28		Physics	UCI
72	gas_drift	13910	128	0	128	6	78.37		−2.04		Computer	UCI
73	gcm_test	46	16063	0	16063	14	86.96		2.54		N/A	OM
74	gesture_phase_segmentation	9873	32	0	32	5	70.12		−2.49		N/A	UCI
75	gina_agnostic	3468	970	0	970	2	49.16		−0.55		Computer	UW

(*continued on next page*)

Table 10 (*continued*).

ID	Dataset	#E	#F	#FQL	#FQT	#C	MCE	?	R	B	D	S
76	gina_prior	3468	784	0	784	2	49.16		−0.65		Computer	UW
77	gina_prior2	3468	784	0	784	10	88.96		−0.65		Computer	UW
78	glass	214	9	0	9	6	64.49		−1.38		Physics	UCI
79	grub_damage	155	8	6	2	4	68.39		−1.29	✓	N/A	UW
80	haberman	306	3	1	2	2	26.47		−2.01	✓	Life	UCI
81	har	10299	561	0	561	6	81.12		−1.26		N/A	OM
82	hayes_roth_train	132	4	0	4	3	61.36		−1.52		N/A	UW
83	heart_c	303	13	7	6	2	45.54	✓	−1.37	✓	Life	UCI
84	heart_h	294	13	7	6	2	36.05	✓	−1.35	✓	Life	UW
85	heart_statlog	270	13	0	13	2	44.44		−1.32		Life	UCI
86	hepatitis	155	19	13	6	2	20.65	✓	−0.91	✓	Life	UCI
87	hepatobiliary_disorders	536	9	0	9	4	66.79		−1.77		N/A	Other
88	higgs	46000	28	0	28	2	46.91		−3.22		N/A	Other
89	hill_valley	1212	100	0	100	2	50.00		−1.08		N/A	UCI
90	hypothyroid	3772	29	22	7	4	7.71	✓	−2.11	✓	Life	UCI
91	internet_advertisements	3279	1558	0	1558	2	14.00		−0.32		Computer	UCI
92	ionosphere	351	34	0	34	2	35.90		−1.01		Physics	UCI
93	iris	150	4	0	4	3	66.67		−1.57		Life	UCI
94	irish_educational_transitions	500	5	3	2	2	44.40	✓	−2.00	✓	N/A	UW
95	isolet	7797	617	0	617	26	96.15		−1.10		Computer	UCI
96	jm1	10885	21	0	21	2	19.35	✓	−2.71		N/A	OM
97	kdd_japanese_vowels_test	5687	14	0	14	9	79.08		−2.61		N/A	UW
98	kdd_japanese_vowels_train	4274	14	0	14	9	85.82		−2.48		N/A	UW
99	kdd_synthetic_control	600	60	0	60	6	83.33		−1.00		N/A	UCI
100	kr_vs_kp	3196	36	36	0	2	47.78		−1.95		Game	UCI
101	labor	57	16	8	8	2	35.09	✓	−0.55	✓	Social	UCI
102	letter	20000	16	0	16	26	95.94		−3.10		Computer	UCI
103	leukemia	72	7129	0	7129	2	34.72		2.00		Life	OM
104	leukemia_3classes	72	7129	0	7129	3	47.22		2.00		Life	UW
105	leukemia_test	34	7129	0	7129	2	41.18		2.32		Life	UW
106	leukemia_train	38	7129	0	7129	2	28.95		2.27		Life	UW
107	lung_cancer	32	56	56	0	3	59.38	✓	0.24		Life	UCI
108	lupus	87	3	0	3	2	40.23		−1.46		Life	OM
109	lymphography	148	18	15	3	4	45.27		−0.91	✓	Life	UCI
110	lymphoma_11classes	96	4026	0	4026	11	76.04	✓	1.62		N/A	OM
111	lymphoma_2classes	45	4026	0	4026	2	48.89	✓	1.95		N/A	OM
112	lymphoma_9classes	96	4026	0	4026	9	52.08	✓	1.62		N/A	OM
113	madelon	2600	500	0	500	2	50.00		−0.72		N/A	UCI
114	magic_telescope	19020	11	0	11	2	35.16		−3.24		Physics	UCI
115	mammographic_masses	961	5	4	1	2	46.31	✓	−2.28	✓	Life	UCI
116	mfeat_factors	2000	216	0	216	10	90.00		−0.97		Computer	UCI
117	mfeat_fourier	2000	76	0	76	10	90.00		−1.42		Computer	UCI
118	mfeat_morphological	2000	6	0	6	10	90.00		−2.52		Computer	UCI
119	mfeat_pixel	2000	240	240	0	10	90.00		−0.92		Computer	UCI
120	mfeat_zernike	2000	47	0	47	10	90.00		−1.63		Computer	UCI
121	mice_protein	1080	81	4	77	8	86.11	✓	−1.12	✓	Life	UCI
122	micro_mass	571	1300	0	1300	20	89.49		0.36		Life	UCI
123	mnist_784	70000	784	0	784	10	88.75		−1.95		Computer	OM
124	molecular_biology_promoters	106	56	56	0	4	67.92		−0.28		Life	UCI
125	monks_problems_1test	432	6	6	0	2	50.00		−1.86		N/A	UCI
126	monks_problems_1train	124	6	6	0	2	50.00		−1.32		N/A	UCI
127	monks_problems_2test	432	6	6	0	2	32.87		−1.86		N/A	UCI
128	monks_problems_2train	169	6	6	0	2	37.87		−1.45		N/A	UCI
129	monks_problems_3test	432	6	6	0	2	47.22		−1.86		N/A	UCI
130	monks_problems_3train	122	6	6	0	2	49.18		−1.31		N/A	UCI
131	mushroom	8124	22	22	0	2	48.20	✓	−2.57		Life	UCI
132	nomao	34465	118	29	89	2	28.56		−2.47	✓	Computer	UCI
133	nursery	12960	8	8	0	5	66.67		−3.21		Social	UCI
134	oh0_wc	1003	3182	0	3182	10	80.66		0.50		Computer	UW
135	oh10_wc	1050	3238	0	3238	10	84.29		0.49		Computer	UW
136	oh15_wc	913	3100	0	3100	10	82.80		0.53		Computer	UW
137	oh5_wc	918	3012	0	3012	10	83.77		0.52		Computer	UW
138	one_hundred_plants_margin	1600	64	0	64	100	99.00		−1.40		Life	UCI
139	one_hundred_plants_shape	1600	64	0	64	100	99.00		−1.40		Life	UCI
140	one_hundred_plants_texture	1599	64	0	64	100	99.00		−1.40		Life	UCI
141	optdigits	5620	64	0	64	10	89.82		−1.94		Computer	UCI
142	ova_breast	1545	10936	0	10936	2	22.27		0.85		Life	OM
143	ova_colon	1545	10936	0	10936	2	18.51		0.85		Life	OM
144	ova_kidney	1545	10936	0	10936	2	16.83		0.85		Life	OM
145	ova_lung	1545	10936	0	10936	2	8.16		0.85		Life	OM
146	ova_omentum	1545	10936	0	10936	2	4.98		0.85		Life	OM
147	ova_ovary	1545	10936	0	10936	2	12.82		0.85		Life	OM
148	ova_uterus	1545	10936	0	10936	2	8.03		0.85		Life	OM
149	ozone_level_8hr	2534	72	0	72	2	6.31		−1.55		Physics	UCI
150	page_blocks	5473	10	0	10	5	10.23		−2.74		Computer	UCI

(*continued on next page*)

Table 10 (*continued*).

ID	Dataset	#E	#F	#FQL	#FQT	#C	MCE	?	R	B	D	S
151	parkinsons	195	21	0	21	2	24.62		−0.97		Life	UCI
152	pasture_production	36	22	1	21	3	66.67		−0.21	✓	N/A	UW
153	pendigits	10992	16	0	16	10	89.59		−2.84		Computer	UCI
154	phishing_websites	11055	30	30	0	2	44.31		−2.57		N/A	OM
155	phoneme	5404	5	0	5	2	29.35		−3.03		N/A	OM
156	pima	768	8	0	8	2	34.90		−1.98		Life	UCI
157	postoperative_patient	90	8	8	0	3	28.89	✓	−1.05		Life	UCI
158	primary_tumor	339	17	17	0	21	75.22	✓	−1.30		Life	UCI
159	prnn_crabs	200	7	1	6	2	50.00		−1.46	✓	N/A	OM
160	qsar_biodeg	1055	41	0	41	2	33.74		−1.41		N/A	UCI
161	red_white_wine	6497	12	1	11	7	56.35		−2.73	✓	N/A	Other
162	satimage	1286	36	0	36	6	76.05		−1.55		Physics	UCI
163	sa_heart	462	9	1	8	2	34.63		−1.71	✓	Life	OM
164	schizo	340	14	2	12	2	47.94	✓	−1.39	✓	Life	OM
165	segment	2310	19	0	19	7	85.71		−2.08		N/A	UCI
166	semeion	1593	256	256	0	10	89.83		−0.79		Computer	UCI
167	shuttle_landing_control	15	6	6	0	2	40.00	✓	−0.40		Physics	UCI
168	sick	3772	29	22	7	2	6.12	✓	−2.11	✓	N/A	UW
169	solar_flare1	323	12	12	0	6	72.76		−1.43		Physics	UW
170	solar_flare2	1066	12	12	0	6	68.95		−1.95		Physics	UW
171	sonar	208	60	0	60	2	46.63		−0.54		Physics	UCI
172	soybean	683	35	35	0	19	86.53	✓	−1.29		Agriculture	OM
173	spambase	4601	57	0	57	2	39.40		−1.91		Computer	UCI
174	spect	267	22	22	0	2	20.60		−1.08		Life	UCI
175	spectf_test	269	44	0	44	2	20.45		−0.79		Life	UW
176	spectf_train	80	44	0	44	2	50.00		−0.26		Life	UW
177	spectrometer	531	101	1	100	48	89.64		−0.72	✓	Physics	UCI
178	spect_test	187	22	22	0	2	8.02		−0.93		Life	UW
179	spect_train	80	22	22	0	2	50.00		−0.56		Life	UW
180	splice	3190	60	60	0	3	48.12		−1.73		Life	UCI
181	sponge	76	44	44	0	3	7.89	✓	−0.24		Life	UCI
182	squash_stored	52	24	3	21	3	55.77	✓	−0.34	✓	N/A	UW
183	squash_unstored	52	23	3	20	3	53.85	✓	−0.35	✓	N/A	UW
184	steel_plates_fault	1941	33	0	33	2	34.67		−1.77		N/A	OM
185	sylva_agnostic	14395	216	0	216	2	6.15		−1.82		Life	UW
186	sylva_prior	14395	108	0	108	2	6.15		−2.12		Life	UW
187	tae	151	5	2	3	3	65.56		−1.48	✓	N/A	UCI
188	tamilnadu_electricity	45781	3	1	2	20	93.65		−4.18	✓	Life	UCI
189	teaching_assistant	151	6	4	2	3	65.56		−1.40	✓	Social	UW
190	tic_tac_toe	958	9	9	0	2	34.66		−2.03		Game	UCI
191	tr11_wc	414	6429	0	6429	9	68.12		1.19		Computer	UW
192	tr12_wc	313	5804	0	5804	8	70.29		1.27		Computer	UW
193	tr23_wc	204	5832	0	5832	6	55.39		1.46		Computer	UW
194	tr31_wc	927	10128	0	10128	7	62.03		1.04		Computer	UW
195	tr41_wc	878	7454	0	7454	10	72.32		0.93		Computer	UW
196	tr45_wc	690	8261	0	8261	10	76.81		1.08		Computer	UW
197	trains	10	32	32	0	2	50.00	✓	0.51		N/A	UCI
198	user_knowledge	403	5	0	5	5	67.99		−1.91		Computer	UCI
199	vehicle	846	18	0	18	4	74.23		−1.67		N/A	OM
200	vertebra_column	310	6	0	6	2	32.26		−1.71		N/A	UCI
201	volcanoes_a3	1521	3	0	3	5	9.99		−2.71		N/A	OM
202	vote	435	16	16	0	2	38.62	✓	−1.43		Social	UCI
203	vowel	990	12	2	10	11	90.91		−1.92	✓	N/A	OM
204	walking_activity	149332	4	0	4	22	85.27		−4.57		N/A	OM
205	wap_wc	1560	8460	0	8460	20	78.14		0.73		Computer	UW
206	waveform	5000	40	0	40	3	66.16		−2.10		Physics	UCI
207	wdbc	569	30	0	30	2	37.26		−1.28		Life	UCI
208	white_clover	63	31	4	27	4	39.68		−0.31	✓	N/A	UW
209	wilt	4839	5	0	5	2	5.39		−2.99		Life	UCI
210	wine	178	13	0	13	3	60.11		−1.14		Physics	UCI
211	wine_quality_white	4898	11	0	11	7	55.12		−2.65		Business	UCI
212	yeast	1484	8	0	8	10	68.80		−2.27		Life	UCI
213	zoo	101	16	15	1	7	59.41		−0.80	✓	Life	UCI

The employment of the meta-feature engineering model resulted in nine Meta-Bases (MB), each composed of 1456 meta-examples described by 161 input meta-features (I) and one target meta-feature (T). The nominal values that T can assume within each meta-base are described below, where BF stands for Best First, GS for Greedy Search, 10 for selecting 10% of the original number of features, 20 for 20%, 40 for 40%, and 80 for selecting 80% of the original number of features.

MB1. $T \in$ {"Filter", "Wrapper"};

MB2. $T \in$ {"FSS", "Ranking"};

MB3. $T \in$ {"CBF", "CFS"};

MB4. $T \in$ {"CBF_{BF}", "CBF_{GS}"};

MB5. $T \in$ {"CFS_{BF}", "CFS_{GS}"};

MB6. $T \in$ {"InfoGain", "ReliefF"};

MB7. $T \in$ {"$InfoGain_{10}$", "$InfoGain_{20}$", "$InfoGain_{40}$", "$InfoGain_{80}$"};

MB8. $T \in$ {"$ReliefF_{10}$", "$ReliefF_{20}$", "$ReliefF_{40}$", "$ReliefF_{80}$"};

MB9. $T \in$ {"WSE_{BF}", "WSE_{GS}"}.

In Step 3, aiming to induce meta-models for feature selection algorithm recommendation, we feed the C4.5, RF, XGBoost, ANN, and SVM metalearners with the nine multiclass MB generated in Step 2. The built meta-models were then organized into a hierarchy or chain, as shown in Fig. 13. In this figure, parent nodes represent classifiers, each built on a given MB, and child nodes indicate either classifiers or responses from classifiers (labels). A new meta-example (dataset) can be tested on the classifiers in a top-down manner, in order to determine the most appropriate feature selection algorithm for the problem at hand. This labeling chain is one of several approaches for suggesting feature selection algorithms. In this work, we limit ourselves to it due to its simplicity and effectiveness.

We evaluated the proposed meta-feature engineering model based on the predictive performance of the meta-models built from the nine MB. To do so, we applied a 10-fold stratified cross-validation.

In Step 4, we analyzed the meta-feature engineering model and its results from three perspectives: (i) meta-base properties, including the proportion between objects and classes; (ii) input meta-feature significance, *i.e.*, how important are the set of characterization measures for metalearning; and (iii) metalearning performance, which covers the accuracy of the metalearners for feature selection algorithm recommendation. To support our discussion, we conducted significance tests using seven statistical methods available at MATLAB[6]: Shapiro–Wilk normality test; Mann–Whitney, Kruskal–Wallis, Unpaired t-test and One-way ANOVA tests with p-value < 0.05; and Friedman test with p-value < 0.05 followed by Tukey–Kramer posthoc test.

All codes associated with our meta-feature engineering model were implemented using R[7] and Java[8]. As for the feature selection algorithms and hybrid ensemble classification methods, we considered the versions available in the Weka library. For the machine learning algorithms chosen as metalearners, we used the versions compatible with the CARET[9] package from R. The metalearners parameter values were estimated via `caret` by means of 5-fold cross-validation and $tuneLength = 5$. We employed default parameter values for all other methods.

6. Results and discussion

We organized the discussion of our experimental results into three main studies: (i) meta-base properties — Section 6.1, (ii) input meta-feature significance — Section 6.2, and (iii) metalearning performance — Section 6.3.

6.1. Meta-base properties

By analyzing Fig. 14 and Table 11, one can observe the influence of the feature selection approaches and settings on meta-base imbalance. For example, the most imbalanced meta-base is MB1, where 96% of the meta-examples share the same label. This finding is partially associated with the large difference between the number of Filter and Wrapper settings evaluated (12 vs. 2, respectively). Thus, a meta-example will be labeled as Wrapper only if a WSE setting outperforms 12 Filter algorithms in the MCPM. Also, the number of features selected by the WSE settings was too small, as at least 51.85% of the subsets contain less than 1% of the features of each dataset. This issue can hinder learning performance when several relevant features are discarded.

An additional property that potentially influences the MB1 imbalance involves the fact that Naive Bayes, the learning algorithm used to evaluate features in Wrapper, assumes class conditional independence, *i.e.*, each feature is independently taken into account as a random variable. This assumption, however, is not considered by three out of the four classification algorithms used in the ensemble inherent to the MCPM. Thus, the features chosen by WSE could lead to worse results than the features selected by the 12 classifier-independent Filter settings, resulting in fewer meta-examples labeled as Wrapper. Nevertheless, investigating Wrapper feature selection based on Naive Bayes, BF, and GS is still relevant. Combinations among these methods were evaluated in different domains, as illustrated in Kohavi and John (1997). Also, some metalearners were able to correctly recommend these combinations in several MB9 meta-examples containing "WSE_{BF}" or "WSE_{GS}" as their target meta-feature values (Section 6.3). The fact that forward search BF and GS include one feature at a time, and the new subset is evaluated by a learner assuming class conditional independence in WSE, can be associated with these results.

Table 11
Percentage of the 1456 datasets in which the percentage of selected features (p) fell within a specific interval. Only FSS algorithms were considered, as the ranking ones were applied with a fixed number of features.

Interval	CBF		CFS		WSE	
	BF	GS	BF	GS	BF	GS
(0%,1%)	21.09	21.43	14.63	14.77	51.85	64.56
[1%,10%)	22.05	17.38	30.01	31.73	17.45	13.39
[10%,20%)	6.87	5.15	20.95	20.33	7.35	6.32
[20%,40%)	8.59	7.55	19.30	18.54	10.92	8.24
[40%,80%)	10.23	7.97	13.53	13.26	9.75	5.98
[80%,100%)	31.18	40.52	1.58	1.37	2.68	1.51

As is the case with MB1, MB2 imbalance could be related to the difference between the number of ranking and filter FSS settings evaluated in this paper (8 vs. 4, respectively). Also, at least 53.50% of the feature subsets chosen by any filter FSS setting are larger than or equal to the ones derived from $InfoGain_{10}$ (Table 11), the leading method according to the MCPM (Table 12), and $ReliefF_{10}$. Moreover, the settings from CBF, an FSS approach, chose 80% or more features from at least 31.18% of the datasets. It should be emphasized that feeding the ensemble with reasonably small subsets, composed of several relevant features, may be helpful to reduce the effects of the "curse of dimensionality" (Liu & Motoda, 2007).

Different from MB3 and MB4, MB5 is relatively imbalanced. The number of settings evaluated is not relevant in this case, as it is the same amount associated with MB4. MB5 also takes into account the same search strategies considered in MB4: BF and GS. Thus, MB5 imbalance may be related to its specific combinations between the FSS algorithm and search strategies. CFS (MB5) can benefit from the backtracking procedure in BF to reduce the chance to stall at a local optimum when searching for subsets with relevant and non-redundant features. CBF (MB4), in turn, can fit well with simpler search strategies, as the consistency of any subset never increases on that of the full set of features.

MB6 is the only meta-base directly comparing two ranking algorithms. Besides standing out in terms of the number of labeled datasets, InfoGain avoids a few issues inherent to ReliefF. Recall that ReliefF is an algorithm based on a distance measure calculated between examples, which in turn is sensitive to the "curse of dimensionality". In fact, two examples close in a space described by few features are likely distant in a space associated with several features (Liu & Motoda, 2007). Also, ReliefF has more parameters to configure, such as the number of neighbors to take into account during importance estimation and the possibility of using a weighting scheme based on the distance between neighbors. It should also be emphasized that, if one compares InfoGain and ReliefF when the same feature subset size is used, $InfoGain_{10}$, $InfoGain_{20}$, and $InfoGain_{40}$ reached the leading MCPM value more often than the ReliefF counterparts (Table 12).

Meta-bases MB7 and MB8 show class distributions that follow a common trend in feature selection: a smaller number of chosen features can lead to classifiers competitive against models built using more

[6] https://www.mathworks.com/products/matlab.html.
[7] https://www.r-project.org.
[8] https://www.java.com.
[9] http://topepo.github.io/caret/index.html.

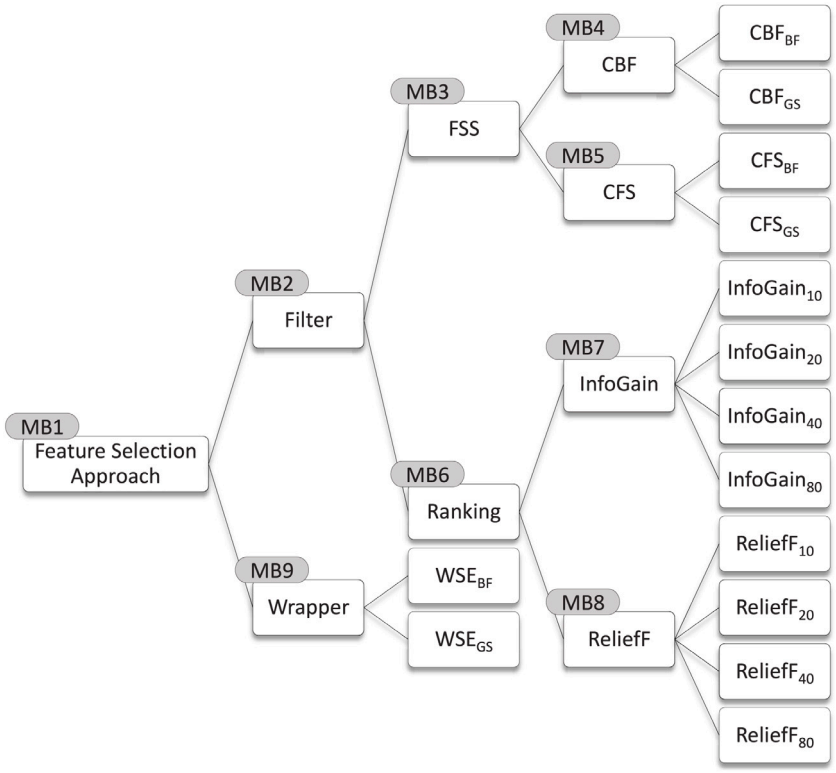

Fig. 13. Hierarchy of multiclass classifiers.

Table 12
Percentage of the 1456 datasets in which a feature selection algorithm reached the best MCPM value.

CBF		CFS		InfoGain				ReliefF				WSE	
BF	GS	BF	GS	10	20	40	80	10	20	40	80	BF	GS
10.85	9.34	3.91	3.30	48.76	4.81	3.16	0.62	7.21	1.58	1.58	1.37	0.41	3.09

features. InfoGain, highlighted against ReliefF in MB6, follows this tendency with more clarity in MB7, as the settings associated with smaller subsets yielded more labeled examples than the remaining settings. One could also find that smaller subsets of features have less redundant features than larger ones, which is useful to build simpler learning models. $ReliefF_{80}$ is the only exception to the trend, as it labels more meta-examples than $ReliefF_{40}$ and $ReliefF_{20}$ in MB8.

As was the case in other meta-bases, we considered the potential influence of properties of the feature selection approaches and settings on the MB9 imbalance. GS does not perform backtracking but stops the search as soon as including the best remaining feature decreases the subset quality, estimated according to the Naive Bayes performance in WSE. BF, in turn, backtracks to reduce the chance to stall at local optima. As both settings were applied for the forward selection, GS could lead to simpler classifiers, with fewer features than BF. In fact, following the trend previously mentioned, (i) WSE_{GS} led to small subsets, with less than 10% of the features of each dataset, more often than WSE_{BF} (Table 11); (ii) WSE_{GS} was considered the best algorithm in terms of the MCPM more often than WSE_{BF} (Table 12). Reasons that could explain why these findings diverge from the ones taken from MB5, another imbalanced meta-base associated with BF and GS, include the use of different feature selection algorithms and approaches and the class conditional independence that WSE assumes.

6.2. Input meta-feature significance

To estimate the descriptors ability to support metalearning for feature selection algorithm recommendation, we conducted statistical tests of significance for each input meta-feature and meta-base. One can consider a meta-base MBZ to illustrate a data table MBZ_s submitted to these tests. MBZ_s takes into account the values of an input meta-feature s to estimate s significance regarding the MBZ target meta-feature. Given the s value describing a dataset d_i, it is included in the table column (group) concerning the corresponding target meta-feature nominal value, *i.e.*, the feature selection algorithm or approach that labels d_i. The resulting table is then submitted to a normality test to verify if it follows the Gaussian distribution. Depending on the result, a parametric or non-parametric test is employed with significance level $\alpha = 0.05$ and the null hypothesis that the s values for distinct MBZ nominal values are significantly similar. Two columns (nominal values) define the tables associated with the binary meta-bases MB1, MB2, MB3, MB4, MB5, MB6 and MB9, while four columns specify the other tables. We used the number of groups in each table to decide the statistical test to apply.

6.2.1. Fine-grained analysis of input meta-features

Five input meta-features described in Table 8, firstly applied by us in the context of feature selection algorithm recommendation, stood out as statistically significant in the nine meta-bases:

- Average strength (weighted graph);
- Correlation degree centrality;
- Maximum local scan;
- Average local scan;
- Median local scan.

These characteristics are variations of three ideas: strength, centrality, and local scan. To discuss them, we recall that, in this work, each vertex in a graph (complex network) represents an example from a

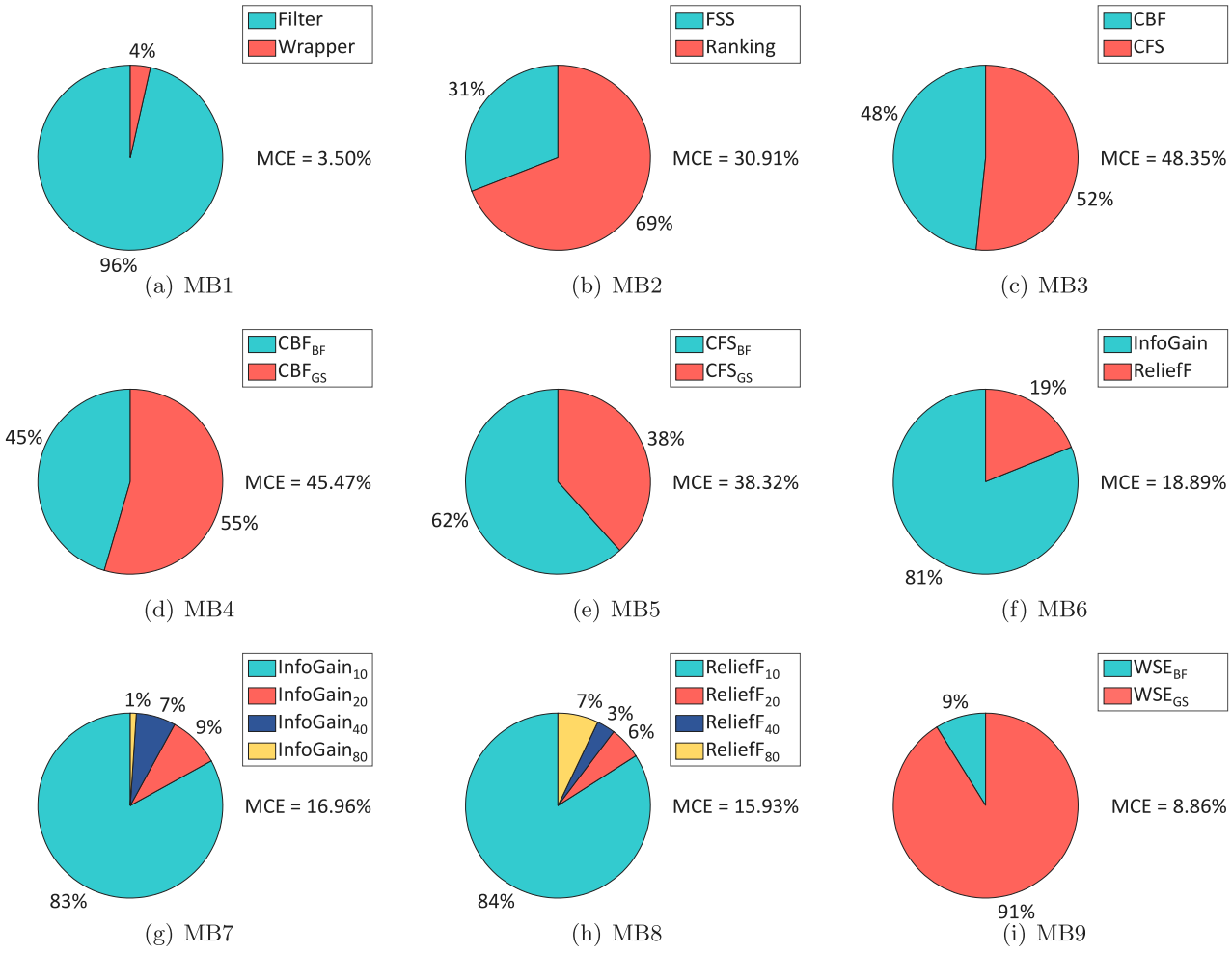

Fig. 14. Target meta-feature distribution in nine meta-bases.

dataset that belongs to a meta-base. In turn, a network is built for each dataset according to the Mutual 10NN algorithm cited in Section 5.2. In the weighted graph version, each edge is weighted by the normalized distance between the connected vertices.

In this scenario, a low average strength suggests that many vertices (examples) are adjacent to edges with small weights (distance between examples). Thus, many examples in the dataset would be close to its mutual neighbors.

A graph with high correlation degree centrality indicates an association between the degree and the centrality of many vertices (examples). In particular, for many examples, the number of neighbors would tend to be related to the subgraph centrality, *i.e.*, the number of simple cycles departing from a vertex.

Local scan takes into account the surroundings of each vertex to estimate network statistics such as the maximum number of edges across all vertex neighborhoods (Priebe et al., 2005). In this study, three statistics highlighted according to the statistical tests: the maximum, the average, and the median number of edges across all vertex neighborhoods of size 1. All measures can be useful, as a dataset with a high value in the corresponding input meta-features has one or more vertices with a large number of mutual neighbors in the network. In this scenario, the existence of many neighbors from the same class of the "central" vertices could ease the selection of features able to separate instances from different classes.

One may note that the instance neighborhood idea associated with network strength, centrality, and local scan is useful for one of the considered feature selection algorithms (ReliefF) and one of the learning algorithms used in the hybrid ensemble to evaluate feature selection (5NN).

Three characterization measures previously highlighted are based on central tendency measures — median and average. Median is a well-known alternative to reduce the effect of extreme values during data central tendency estimation. Average, in turn, gives us a reasonable estimate when conditions such as symmetric distribution hold.

Beyond the five graph-based input meta-features previously discussed, two descriptors that also performed well in the literature (Parmezan et al., 2017) were considered significant in all meta-bases: Signal/noise ratio (Table 1) and Dispersion of the dataset (Table 4). They consider issues potentially relevant for some feature selection algorithms, such as the average mutual information between classes and attributes, and the ratio between the number of examples and the number of attributes in a dataset.

The previous analysis was useful, as taking into account only the average values and corresponding standard deviation of the input meta-features would make it hard to find the most significant descriptors out of the 161 ones. Fig. 15 shows, for the meta-base MB3, the average and the standard deviation values of the 161 input meta-features used in this paper. We chose this meta-base, as an example, because it is the most balanced meta-base (Fig. 14) and illustrates some findings also identified in other meta-bases. The characterization measures are sorted in the chart according to the same order indicated in Table A.1 of the supplementary material.

One can note that the standard deviation is relatively low in comparison with the average values. Moreover, as was the case in the

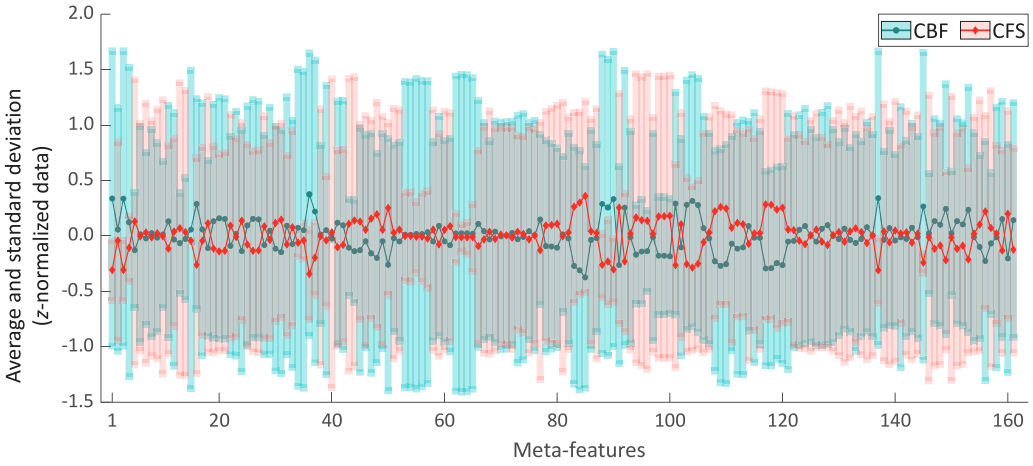

Fig. 15. Average and standard deviation of the values from the 161 input meta-features in MB3.

previous work (Parmezan et al., 2017), several descriptors show similar average values for both classes.

6.2.2. Coarse-grained analysis of input meta-feature categories

We also analyzed significance in a coarser level: input meta-feature category. Table 13 shows three values for each category:

Average % and standard deviation: The percentage of input meta-features that were considered significant for each meta-base, averaged across the nine meta-bases, and the corresponding standard deviation. Each percentage of input meta-features, in turn, is a ratio between the number of significant meta-features and the total number of meta-features per category;

Frequency: The percentage of input meta-features that were deemed significant in at least seven out of the nine meta-bases. We chose seven as a threshold because it represents nearly 80% of the meta-bases evaluated in this study.

The category based on graphs, firstly applied by us for feature selection algorithm recommendation, reached the second best average % and standard deviations. Moreover, five out of the seven descriptors highlighted in the fine-grained analysis (Section 6.2.1) belong to this category. These findings are encouraging, as complex networks are an active research area and several measurements to describe these graphs have been developed (Costa et al., 2007). Thus, a future research direction involves implementing more valuable graph-based measurements as input meta-features and evaluating their ability to support metalearning. By doing so, the frequency associated with this category in Table 13 might increase.

Simple and complexity measures were also highlighted by reaching, respectively, the best average % and the best standard deviation in Table 13. The simple category also outperformed the other ones in terms of frequency. These findings indicate that the simple and complexity categories contain many descriptors that yield values significantly different, for distinct target meta-feature nominal values, in several meta-bases. One can also note that the literature considered both categories as important ones (Parmezan et al., 2017). It should be emphasized that this work extended them by developing new input meta-features for feature selection algorithm recommendation. Two proposed complexity descriptors, the Error rate of the nearest neighbor classifier and Ratio measure, were found significant in eight out of nine meta-bases.

By decreasing the threshold value used by the frequency in Table 13 down to 5, nearly 50% of the statistical tests conducted per input meta-feature, one would note that all statistical, landmarking, and image-based measures were significant in 5 or more meta-bases. These categories also achieved relatively high average % in the same table. This positive finding enhances the contribution of this paper, which firstly applied the image-based category for feature selection algorithm recommendation.

To supplement the previous analysis, we identified, for each category, the five descriptors that reached the highest number of significant differences in the nine statistical tests conducted per input meta-feature. Although six is the number of characterization measures in the smallest category (landmarking), we chose five because it reduced the number of ties among descriptors in terms of the number of differences. Otherwise, only in the graph-based category, 27 input meta-features would tie for second place. Fig. 16 depicts the number of significant differences reached by the top-five characterization measures reported in Table 14.

This study pioneered the application of the best category for feature selection algorithm recommendation shown in Fig. 16: graph-based. Five out of the seven input meta-features highlighted in Section 6.2.1, which also appear in Table 14, belong to this category. It should also be emphasized that the graph-based category, which concentrates the largest number of descriptors in this work, reached at least one significant difference in all of its 82 characterization measures.

Although the model-based category does not contain any input meta-feature considered significant across all meta-bases, its five representatives in Fig. 16 consistently reached eight significant differences. This finding supplements the previous ones by highlighting another category the literature has promoted (Filchenkov & Pendryak, 2015). The investigated model-based measures describe the structure of a decision tree built from each dataset (Peng et al., 2002). As this classifier is a symbolic one, the meaning of the values associated with each input meta-feature is clear.

Other categories previously discussed in this section due to the positive results in one or more criteria also appear with many significant differences in Fig. 16.

6.3. Metalearning performance

We evaluated five metalearners on each of the nine meta-bases considering different combinations of input meta-features. We estimated the quality of the induced meta-models using 10-fold stratified cross-validation and compared the resulting accuracy using the Friedman statistical test with the Tukey–Kramer posthoc test ($p < 0.05$).

Tables 15 and 16 report, for each meta-base formed by a given set of input meta-features, the average of the metalearners' accuracy with their respective standard deviations in parentheses. In these tables, gray cells represent the best average hit rate, *i.e.*, the one that occupied the first position in the Friedman rank when we compared different

Table 13
Average percentage (%), standard deviation and frequency regarding significant meta-features in each category.

	Simple	Statistical	Information	Complexity	Landmarking	Model-based	Image-based	Graph-based
Average %	0.78	0.74	0.72	0.70	0.76	0.73	0.75	0.76
Standard deviation	0.19	0.20	0.28	0.13	0.22	0.25	0.17	0.14
Frequency	0.75	0.42	0.50	0.50	0.67	0.53	0.71	0.63

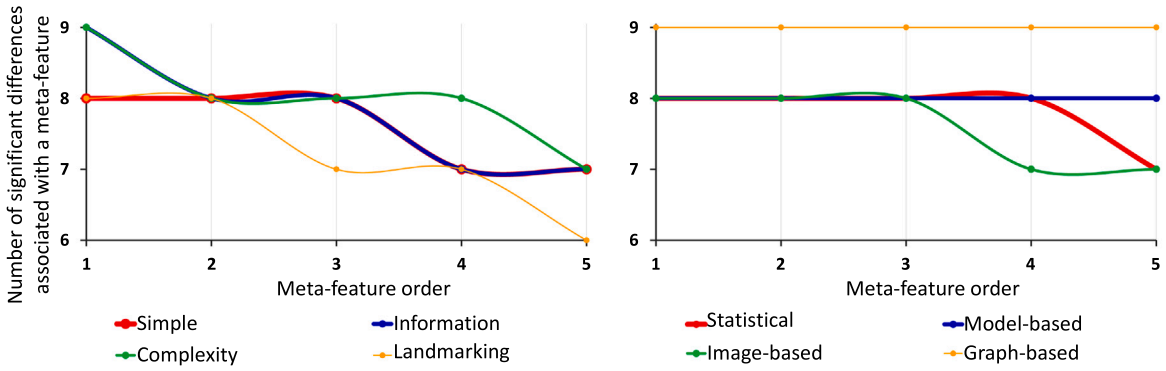

Fig. 16. Number of significant differences associated with an input meta-feature in descending order: top-five input meta-features in each category.

meta-models built using the same metalearner. The inverted triangle (▼) denotes that the meta-model error is higher than the MCE; the asterisk (*) shows that the meta-model's performance considering a specific set of input meta-features presented statistical significant difference concerning the others; average rank comprises the average position that the accuracy of the meta-models induced from the same set of input meta-features occupied in the Friedman rank; and global average rank is the position averaged across all the Friedman ranks.

High hit rates from metalearners are used here as an indication of which descriptors have high explanatory power and, therefore, better describe the problem of feature selection algorithm recommendation. In other words, the sets of input meta-features that best and worst described a meta-base occupied, in this order, the first and last positions in the average rank.

We can see in Table 15, looking at each meta-base described by a set of input meta-features from a given category of dataset characterization measures, that descriptors from the simple category had high predictive power in four meta-bases: MB1, MB4, MB6, and MB8. Input meta-features based on information theory described MB3 and MB7 well. Characterization measures from the complexity category explained MB2 and MB5 better. Graph-based descriptors exhibited high discriminative power in MB9. On the other hand, model-based meta-features provided the poorest results for four meta-bases: MB2, MB3, MB4, and MB5. Landmarkers were not as appropriate to describe MB6 and MB8, just as the image-based characterization measures were not so explanatory in MB1 and MB7. Descriptors from the simple category represented MB9 in a median way.

The most difficult meta-bases to be predicted by metalearners were those involving filter FSS algorithms (MB3, MB4, and MB5). Although MB3 and MB4 are balanced meta-bases, the meta-models built on them presented relatively low average performances regarding the others. A similar finding was noted from relatively imbalanced MB5.

Most of the models built from MB7 and MB9, which are imbalanced meta-bases, presented a classification error higher than the MCE. It means that distinguishing between the performances of different configurations of the InfoGain (MB7) and WSE (MB9) algorithms is a complicated task.

For each meta-base and regardless of metalearner, the average accuracy was very close to each other, except for MB2 and MB3 described by input meta-features from the model-based category and MB3 formed by input meta-features extracted using landmarkers. According to Friedman's average rank, such exceptions provided inferior results with statistically significant differences.

In line with the global average rank, graph-based measures are consistent in describing the problem of feature selection algorithm recommendation despite not showing statistically significant differences concerning the descriptors of the other categories.

Analyzing each meta-base described by one of the six sets of input meta-features in Table 16, we can see that the direct characterization measures had a high predictive power for two meta-bases: MB1 and MB5. Direct characterization input meta-features combined with the complexity ones best described MB3, MB4, and MB7. Direct characterization measures, together with landmarking and based on models, explained MB2 better. While all the descriptors (all the input meta-features) exhibited reasonable discriminative power in MB6 and MB8, the top-five input meta-features (Table 14) provided the best results for MB9. In contrast, direct characterization measures did not explain MB4 so well. The new input meta-features were moderately representative in five meta-bases: MB1, MB2, MB6, MB7, and MB9. The use of all descriptors (all the input meta-features) was not appropriate to predict the problems represented by MB3 and MB5, and the top-five input meta-features resulted in the poorest results for MB8.

Again, the most difficult meta-bases to be predicted by metalearners were those related to filter FSS algorithms (MB3, MB4, and MB5). Besides, most models induced from MB7, which describe the performance of different InfoGain configurations and is an imbalanced meta-base, exhibited a classification error higher than the MCE.

The input meta-features from the combination of direct characterization and complexity categories stood out for having described meta-bases covering FSS algorithms well (MB3, MB4, and MB5). At the same time, all the measures were considered relevant to represent meta-bases involving the performance of algorithms that rank features (MB6, MB7, and MB8).

In agreement with the global average rank, no combination of characterization measure sets exhibited significant differences. In this way, we suggest the use of the top-five input meta-features since this is a diversified set of descriptors that can provide, with a low computational cost, similar or better results than the other characterization measure sets employing a smaller number of input meta-features.

As for the metalearners adopted in this study, we can see in Tables 15 and 16 that no learning algorithm outperformed the others by a large margin of error.

Table 14
Top-five input meta-features in each category highlighted in Fig. 16.

Category	Meta-features in descending order		Number of differences
Simple	1.	Number of qualitative attributes & Number of quantitative attributes & Number of examples	8
	2.	Number of attributes & Number of attributes with missing values & Number of examples with missing values	7
Statistical	1.	Minimum coefficient of variation of the attributes & Average coefficient of variation of the attributes & Average correlation between attributes & Balancing of the dataset	8
	2.	Majority class error	7
Information	1.	Signal/noise ratio	9
	2.	Maximum conditional entropy between classes and attributes & Equivalent number of attributes	8
	3.	Class entropy & Minimum conditional entropy between classes and attributes & Average conditional entropy between classes and attributes	7
Complexity	1.	Dispersion of the dataset	9
	2.	Maximum Fisher's discriminant ratio & Error rate of the nearest neighbor classifier & Ratio measure	8
	3.	Intra-class distance & Attribute class correlation	7
Landmarking-based	1.	Randomly chosen node learner & Linear discriminant	8
	2.	Decision node learner & Elite 1NN learner	7
	3.	Worst node learner	6
Model-based	1.	Tree height & Number of nodes & Longest branch & Smallest branch & Mean branch length & Maximum occurrence of attributes & Mean occurrence of attributes & Standard deviation of the occurrence of attributes	8
Image-based	1.	Energy & Correlation & Entropy	8
	2.	Contrast & Variance & Sum variance & Sum entropy & Difference variance & Information measure of correlation II & Maximal correlation coefficient	7
Graph-based	1.	Average strength (weighted graph)& Correlation degree centrality & Maximum local scan & Average local scan & Median local scan	9

This paper focused on proposing a meta-feature engineering model to represent the problem of the automatic choice of feature selection algorithms. We used the metalearning framework for feature selection algorithm recommendation of Fig. 3 as a reference to design our proposal (Fig. 7), which enables the elaboration of meta-bases for the same task.

We can apply supervised machine learning algorithms on meta-bases to induce algorithm recommendation models (meta-models). As reported in Section 3, three approaches are typically applied for this purpose (Kalousis, 2002): finding the best algorithm or subset, and ranking of algorithms. These manners of suggestion cannot be seen as parameters of a framework, since their codification depends on how the metalearning problem is formulated. For example, multiclass classification: best algorithm; (hierarchical) multi-label classification: subset of algorithms; or label ranking: ranking of algorithms.

As there are different ways of conceiving a meta-model, our work treated the feature selection algorithm recommendation task as a chain of binary/multiclass classifiers. Through this simple yet powerful approach, we built seven binary (MB1, MB2, MB3, MB4, MB5, MB6, and MB9) and two multiclass (MB7 and MB8) classifiers using input meta-features from one (Table 15) and multiple (Tables 16) categories to identify those that best describe the addressed metalearning problem.

Although the present study does not focus on the way of suggesting feature selection algorithms, we evaluated two other important scenarios to quantify our meta-feature engineering model's overall performance. The first scenario comprises a meta-base named baseline that differs from those previously analyzed in terms of the target meta-feature, formed by 12 feature selection algorithms: the tree leaf labels in Fig. 13. The elaboration of such a meta-base involved computing the ground truth, obtained in agreement with the MCPM total area values, for each of the 1456 datasets (Table A.2 of the supplementary material). The second scenario corresponds to the chaining of MB1 with MB2, MB3, MB4, MB5, MB6, MB7, MB8 and MB9 (all the meta-bases), as schematized in Fig. 13 and described in Sections 4.3 and 5.2. For evaluating these scenarios that seek to recommend the best feature selection

Table 15
Hit rates of metalearners to predict feature selection algorithms from sets of input meta-features arranged in eight categories.

Meta-base	Metalearner	Input meta-features							
		Simple	Statistical	Information	Complexity	Landmarking	Model-based	Image-based	Graph-based
MB1	C4.5	97.39 (2.08)	97.32 (2.07)	97.18 (1.99)	97.18 (2.33)	97.32 (2.09)	97.05 (2.20)	96.98 (1.98)▼	97.32 (2.14)
	RF	97.39 (2.08)	97.11 (2.15)	97.46 (2.13)	97.11 (2.13)	97.39 (2.34)	97.32 (2.12)	96.84 (1.70)▼	97.25 (2.23)
	XGBoost	97.32 (2.04)	97.18 (2.33)	97.18 (2.17)	96.77 (2.37)▼	97.11 (2.43)	97.25 (2.13)	97.04 (1.95)	97.25 (2.23)
	ANN	96.49 (1.64)▼	96.49 (1.64)▼	96.63 (1.71)▼	96.49 (1.64)▼	96.49 (1.64)▼	96.49 (1.64)▼	96.49 (1.64)▼	96.49 (1.64)▼
	SVM	97.18 (2.24)	97.25 (2.08)	97.32 (2.14)	97.32 (2.14)	97.18 (2.19)	97.39 (2.08)	97.11 (2.45)	97.39 (2.08)
Average rank		4.17	4.56	4.29	4.82	4.41	4.30	5.19	4.26
MB2	C4.5	91.14 (2.03)	91.69 (2.09)	90.04 (2.52)	91.41 (1.36)	86.26 (2.40)	75.07 (2.19)	89.28 (2.14)	90.93 (1.90)
	RF	90.93 (2.24)	91.62 (2.07)	89.90 (2.54)	91.35 (2.29)	87.43 (1.93)	74.52 (2.83)	89.77 (1.90)	90.87 (2.01)
	XGBoost	91.21 (1.65)	91.21 (1.77)	89.63 (1.94)	91.00 (1.91)	86.19 (2.60)	74.31 (3.37)	89.22 (1.95)	90.80 (1.77)
	ANN	84.95 (7.40)	68.47 (1.95)▼	84.54 (2.40)	82.41 (4.91)	83.03 (1.73)	70.19 (2.76)	61.78 (15.93)▼	73.91 (6.79)
	SVM	89.77 (1.90)	89.83 (2.15)	88.32 (2.48)	89.49 (2.05)	83.86 (2.61)	69.92 (2.63)	89.56 (1.76)	87.36 (1.66)
Average rank		2.84	3.38	4.21	2.78	5.92	7.61*	5.07	4.19
MB3	C4.5	74.45 (2.15)	75.14 (1.89)	75.21 (2.33)	76.45 (3.47)	69.57 (3.66)	63.26 (4.51)	74.59 (2.21)	77.06 (2.53)
	RF	78.64 (2.71)	77.34 (2.42)	77.00 (2.60)	77.54 (2.25)	73.15 (3.48)	65.67 (3.29)	77.33 (2.82)	77.82 (2.45)
	XGBoost	77.61 (2.69)	76.52 (2.23)	76.17 (1.95)	78.03 (2.07)	72.81 (3.20)	64.77 (1.96)	76.44 (2.34)	77.68 (2.71)
	ANN	58.44 (7.59)	61.24 (5.90)	69.10 (1.36)	57.30 (9.16)	63.54 (5.05)	57.22 (5.70)	53.03 (2.86)	63.33 (5.37)
	SVM	73.29 (3.41)	72.60 (1.79)	74.32 (1.79)	74.18 (1.78)	69.44 (3.28)	64.49 (3.72)	76.72 (2.29)	72.73 (2.21)
Average rank		4.04	4.18	3.33	3.69	5.76*	7.47*	4.13	3.40
MB4	C4.5	71.77 (4.26)	71.97 (2.13)	65.94 (4.77)	70.80 (3.95)	63.81 (2.56)	59.89 (6.64)	65.25 (3.01)	70.80 (3.50)
	RF	74.38 (2.01)	72.60 (3.01)	69.30 (3.07)	74.65 (2.68)	68.75 (3.52)	61.33 (3.62)	68.26 (4.03)	72.32 (3.47)
	XGBoost	73.56 (2.13)	72.38 (3.09)	69.16 (3.08)	72.11 (3.19)	65.25 (4.00)	61.46 (3.83)	67.78 (3.16)	71.43 (2.74)
	ANN	53.43 (6.94)▼	52.96 (5.20)▼	60.30 (4.51)	55.90 (5.09)	63.67 (2.67)	59.13 (3.69)	53.66 (4.64)▼	58.79 (7.07)
	SVM	72.87 (2.00)	71.84 (2.37)	66.35 (2.48)	71.56 (3.31)	65.80 (2.75)	59.75 (4.16)	68.55 (3.82)	70.46 (4.07)
Average rank		2.90	3.49	5.03	3.27	5.38	6.85	5.56	3.52
MB5	C4.5	66.48 (2.61)	68.61 (4.76)	63.59 (3.16)	66.00 (2.35)	64.14 (4.48)	63.05 (3.12)	63.11 (3.47)	66.41 (3.14)
	RF	64.83 (4.09)	66.89 (5.62)	65.18 (2.38)	67.17 (3.45)	66.00 (3.75)	62.15 (4.39)	66.14 (5.17)	66.34 (4.86)
	XGBoost	65.73 (4.27)	67.17 (4.39)	66.00 (2.14)	66.76 (2.61)	64.01 (4.54)	64.62 (3.63)	64.97 (4.48)	67.37 (3.79)
	ANN	63.04 (2.68)	61.05 (3.81)▼	61.47 (4.03)▼	62.35 (4.11)	63.73 (3.58)	61.81 (3.61)	61.67 (3.65)▼	62.91 (4.47)
	SVM	66.13 (3.40)	66.82 (3.28)	65.16 (4.07)	66.75 (4.29)	63.38 (6.35)	64.06 (5.11)	64.83 (3.58)	66.41 (4.69)
Average rank		4.32	3.48	5.22	3.47	4.96	5.88	5.12	3.55
MB6	C4.5	83.58 (1.50)	84.27 (2.37)	84.34 (2.55)	83.11 (2.33)	81.11 (2.12)	81.11 (2.12)	83.31 (2.39)	84.20 (2.19)
	RF	84.54 (1.87)	83.17 (2.51)	83.38 (2.63)	84.34 (2.47)	79.26 (2.96)	80.43 (2.57)▼	83.66 (1.91)	84.34 (1.32)
	XGBoost	84.27 (1.83)	83.79 (2.05)	83.86 (2.35)	83.17 (2.62)	80.84 (2.20)▼	81.25 (2.04)	83.39 (2.56)	83.51 (1.00)
	ANN	81.11 (2.12)	81.11 (2.12)	81.11 (2.34)	81.11 (2.12)	81.11 (2.12)	81.11 (2.12)	81.04 (2.12)▼	81.11 (2.12)
	SVM	84.41 (2.45)	82.35 (1.62)	82.96 (2.65)	82.76 (1.19)	81.11 (2.12)	81.32 (2.31)	81.04 (2.12)▼	82.14 (2.02)
Average rank		3.24	3.80	3.75	4.12	6.46	6.05	4.75	3.83
MB7	C4.5	82.90 (2.98)▼	83.59 (3.49)	82.97 (3.67)▼	82.21 (2.45)▼	82.56 (2.83)▼	82.70 (3.29)▼	81.11 (2.70)▼	83.52 (3.23)
	RF	82.63 (2.28)▼	82.35 (2.25)▼	83.65 (2.55)	83.38 (2.82)	81.94 (3.21)▼	81.33 (3.13)▼	81.53 (2.87)▼	83.24 (2.12)
	XGBoost	81.66 (2.78)▼	82.77 (3.14)▼	83.93 (2.33)	81.25 (3.02)▼	82.56 (2.56)▼	82.91 (3.41)▼	82.77 (2.63)▼	82.90 (2.92)▼
	ANN	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)
	SVM	82.97 (3.31)▼	82.90 (3.36)▼	83.79 (2.63)	83.04 (3.28)	82.97 (3.31)▼	83.04 (3.28)	82.97 (3.29)▼	83.11 (2.83)
Average rank		4.89	4.34	3.45	4.71	4.79	4.83	5.06	3.93
MB8	C4.5	86.61 (2.51)	86.06 (2.47)	86.13 (1.58)	85.72 (2.07)	83.52 (1.47)	84.34 (1.98)	84.47 (2.26)	85.52 (2.42)
	RF	86.96 (1.78)	86.82 (1.96)	86.20 (1.48)	85.86 (1.97)	83.59 (2.36)	84.34 (1.95)	84.61 (1.38)	85.44 (1.92)
	XGBoost	86.13 (1.92)	86.75 (2.27)	86.20 (1.72)	85.92 (2.25)	84.41 (1.70)	84.00 (1.66)▼	84.27 (1.75)	86.13 (1.29)
	ANN	84.07 (1.97)	84.07 (1.97)	83.66 (2.04)▼	84.07 (1.97)	84.07 (1.97)	84.07 (1.97)	84.07 (1.97)	84.07 (1.97)
	SVM	84.00 (2.03)▼	84.35 (1.98)	85.03 (1.67)	85.03 (2.01)	83.87 (1.56)	83.87 (2.38)	83.38 (1.72)	84.82 (1.85)
Average rank		3.58	3.60	3.62	3.90	5.97	5.60	5.65	4.08
MB9	C4.5	90.73 (3.04)▼	91.28 (2.83)	90.73 (2.89)▼	90.80 (2.99)▼	91.14 (3.02)	91.21 (2.98)	91.14 (3.02)	91.41 (3.16)
	RF	91.28 (2.96)	91.00 (3.40)▼	90.79 (2.48)▼	91.69 (2.87)	90.93 (3.05)▼	91.21 (2.97)	89.97 (2.62)▼	91.48 (2.59)
	XGBoost	91.07 (3.08)▼	91.62 (3.06)	91.00 (3.51)▼	91.35 (3.08)	91.21 (2.99)	91.21 (3.46)	91.28 (2.55)	91.42 (2.93)
	ANN	91.14 (3.02)	91.28 (3.12)	91.00 (3.07)▼	91.14 (3.02)	91.14 (3.02)	91.00 (2.99)▼	91.14 (3.02)	91.14 (3.02)
	SVM	91.14 (3.02)	91.21 (2.99)	91.48 (3.22)	91.07 (3.07)▼	91.28 (3.12)	91.14 (3.02)	91.21 (2.81)	91.14 (3.02)
Average rank		4.96	4.16	4.71	4.40	4.38	4.65	4.86	3.88
Global average rank		3.88	3.89	4.18	3.91	5.34	5.92	5.04	3.85

algorithm based on dataset characteristics, we performed a 10-fold stratified cross-validation experiment using only the RF classification algorithm which proved to be an excellent metalearner candidate (Tables 15 and 16). As a result, we obtained a multiclass classifier induced on the baseline and classifier chains built from the association of all nine meta-bases. Table 17 displays the average accuracy of the decision models with their respective standard deviations in parentheses for each scenario formed by a given set of input meta-features.

The predictive performances listed in Table 17 reveal that the proposed meta-feature engineering model is promising to represent the feature selection algorithm recommendation task, in particular allowing the classifier chain approach advocated in this work to reach around 90% accuracy using the minimum set of input meta-features found (top-five input meta-features). This average hit rate is equivalent to a gain of 27% over the baseline scenario that resides in the simplest way to suggest algorithms in metalearning. We must emphasize that the

Table 16
Hit rates of metalearners to predict feature selection algorithms by combining sets of input meta-features.

Meta-base	Metalearner	Input meta-features					
		Direct	Direct + Complexity	Direct + Landmarking + Model-based	New input meta-features	All the input meta-features	Top-five input meta-features
MB1	C4.5	97.39 (2.08)	97.18 (2.26)	96.84 (2.16)	97.32 (2.14)	96.77 (2.09)	96.98 (2.32)
	RF	97.11 (2.18)	97.25 (2.39)	96.91 (2.11)	97.32 (2.04)	97.32 (2.12)	97.04 (1.98)
	XGBoost	97.04 (2.21)	96.91 (2.33)	97.25 (2.45)	97.11 (2.03)	97.11 (2.15)	97.04 (2.25)
	ANN	96.49 (1.64)▾	96.49 (1.64)▾	96.49 (1.64)▾	96.49 (1.64)▾	96.49 (1.64)▾	96.49 (1.64)▾
	SVM	97.32 (2.14)	97.32 (2.14)	97.32 (2.14)	96.29 (1.60)▾*	97.39 (2.08)	97.32 (2.14)
Average rank		3.32	3.47	3.60	3.70	3.36	3.55
MB2	C4.5	91.48 (2.22)	91.07 (2.18)	91.76 (3.05)	90.79 (2.38)	91.41 (2.13)	91.27 (1.73)
	RF	91.69 (2.16)	91.69 (1.57)	91.69 (2.17)	91.21 (2.17)	92.03 (1.99)	91.69 (2.51)
	XGBoost	91.69 (2.14)	91.76 (2.70)	92.24 (2.12)	91.21 (1.86)	91.55 (2.36)	91.55 (2.64)
	ANN	68.61 (2.48)▾	67.31 (2.51)▾	66.69 (1.98)▾	80.41 (4.70)*	68.27 (3.09)▾	67.51 (3.05)▾
	SVM	91.07 (2.42)	91.07 (1.91)	91.96 (2.57)	86.53 (2.08)	90.04 (1.93)	90.80 (1.69)
Average rank		3.38	3.71	3.03	3.76	3.56	3.56
MB3	C4.5	76.92 (2.11)	77.61 (1.81)	78.09 (2.86)	76.38 (3.54)	75.90 (3.36)	77.20 (2.55)
	RF	78.03 (2.72)	78.37 (1.81)	78.51 (2.77)	77.68 (2.57)	78.38 (2.88)	79.34 (2.65)
	XGBoost	76.45 (3.22)	77.54 (1.16)	77.20 (2.40)	77.41 (3.21)	77.89 (3.08)	77.06 (3.03)
	ANN	63.18 (2.47)	63.11 (2.26)	62.56 (2.81)	63.59 (2.36)	57.60 (6.44)	59.47 (5.14)
	SVM	75.28 (2.88)	76.45 (2.10)	76.79 (2.54)	69.23 (2.81)	72.18 (2.65)	76.79 (1.57)
Average rank		3.49	2.92	3.06	4.06	4.19	3.28
MB4	C4.5	72.05 (2.57)	72.39 (3.44)	71.77 (2.84)	71.56 (3.18)	71.14 (3.80)	73.08 (3.11)
	RF	72.52 (2.81)	74.31 (2.96)	73.63 (2.38)	73.90 (2.83)	74.52 (1.54)	74.24 (1.82)
	XGBoost	72.73 (2.35)	72.31 (4.01)	72.53 (3.03)	72.25 (2.28)	73.90 (1.65)	73.49 (1.43)
	ANN	52.60 (5.01)▾	54.96 (4.34)	54.80 (3.67)	60.98 (5.55)	51.85 (5.30)▾	50.26 (4.51)▾
	SVM	71.42 (2.93)	73.28 (2.27)	72.73 (2.00)	69.15 (5.33)	69.98 (3.28)	71.36 (2.37)
Average rank		3.96	3.16	3.36	3.54	3.54	3.44
MB5	C4.5	68.34 (4.53)	67.03 (3.74)	65.80 (1.86)	66.41 (5.80)	67.17 (4.34)	66.34 (4.85)
	RF	68.00 (3.91)	67.78 (4.22)	66.27 (5.31)	66.89 (4.64)	66.75 (5.98)	66.68 (4.15)
	XGBoost	67.44 (3.75)	67.23 (3.66)	66.48 (4.11)	68.27 (4.31)	67.38 (5.42)	67.92 (3.85)
	ANN	61.80 (3.78)	61.60 (3.67)▾	61.80 (3.55)	60.98 (3.74)▾	61.74 (3.49)	58.04 (6.52)▾
	SVM	66.75 (3.40)	66.07 (3.14)	65.85 (4.87)	66.75 (4.23)	66.21 (2.94)	66.13 (3.59)
Average rank		2.94	3.42	3.75	3.50	3.82	3.57
MB6	C4.5	84.55 (2.09)	84.27 (1.23)	83.93 (2.31)	84.48 (1.85)	84.62 (2.02)	83.86 (2.15)
	RF	83.86 (1.94)	84.48 (2.42)	84.34 (2.27)	83.45 (1.92)	84.48 (2.31)	84.34 (2.81)
	XGBoost	84.82 (1.64)	84.41 (2.27)	84.48 (2.25)	83.04 (1.49)	83.72 (2.97)	84.48 (2.92)
	ANN	81.11 (2.12)	81.11 (2.12)	81.11 (2.12)	81.11 (2.12)	81.11 (2.12)	81.11 (2.12)
	SVM	83.79 (2.00)	83.52 (1.89)	83.79 (2.36)	79.67 (1.35)▾	84.61 (2.43)	83.24 (1.69)
Average rank		3.31	3.39	3.40	4.34	3.06	3.50
MB7	C4.5	81.53 (2.51)▾	82.42 (2.70)▾	81.73 (3.55)▾	81.72 (2.64)▾	82.01 (3.72)▾	80.77 (2.93)▾
	RF	82.97 (2.20)▾	83.52 (3.14)	82.90 (2.74)▾	82.69 (2.83)▾	83.24 (2.72)	82.76 (2.80)▾
	XGBoost	83.45 (2.81)	82.21 (2.66)	83.31 (3.27)	83.04 (3.15)	83.03 (3.05)▾	83.45 (3.32)
	ANN	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)	83.04 (3.28)
	SVM	82.97 (3.31)▾	83.04 (2.77)	82.56 (2.63)▾	80.98 (3.24)▾	82.56 (2.89)▾	83.11 (2.80)
Average rank		3.48	3.26	3.45	4.15	3.30	3.36
MB8	C4.5	85.79 (2.49)	86.21 (1.90)	86.75 (2.28)	86.13 (2.54)	86.75 (2.18)	86.69 (1.94)
	RF	87.04 (2.41)	86.96 (1.96)	86.97 (3.01)	86.88 (1.92)	86.82 (2.26)	86.62 (2.40)
	XGBoost	86.55 (2.42)	86.42 (2.78)	85.80 (2.70)	86.88 (1.80)	86.61 (1.94)	86.06 (2.33)
	ANN	83.94 (2.10)▾	83.86 (1.98)▾	84.14 (2.01)	84.07 (1.97)	84.28 (2.16)	84.00 (2.14)▾
	SVM	84.07 (1.91)	84.35 (2.00)	84.76 (2.35)	84.35 (1.94)	84.55 (1.74)	84.28 (1.96)
Average rank		3.63	3.50	3.38	3.47	3.24	3.78
MB9	C4.5	91.48 (2.68)	91.14 (3.75)	91.83 (3.52)	91.00 (3.17)▾	91.48 (2.68)	91.48 (3.06)
	RF	91.14 (3.14)	91.34 (2.74)	91.48 (2.87)	91.28 (3.13)	91.28 (3.19)	91.41 (3.17)
	XGBoost	91.82 (2.98)	90.73 (2.93)▾	91.00 (3.77)▾	91.28 (2.74)	91.62 (3.58)	91.83 (3.15)
	ANN	91.00 (2.99)▾	91.14 (3.06)	91.07 (3.05)▾	91.14 (3.02)	91.14 (3.02)	91.14 (3.07)
	SVM	91.55 (3.04)	91.76 (2.99)	91.76 (3.01)	89.01 (3.34)▾*	91.55 (2.20)	91.62 (2.58)
Average rank		3.35	3.59	3.27	4.30	3.28	3.21
Global average rank		3.43	3.38	3.37	3.87	3.48	3.47

Table 17
Hit rates of (chained) multiclass classifiers to predict feature selection algorithms by combining sets of input meta-features.

Meta-base	Metalearner	Input meta-features					
		Direct	Direct + Complexity	Direct + Landmarking + Model-based	New input meta-features	All the input meta-features	Top-five input meta-features
Baseline	RF	62.51 (3.86)	61.75 (3.33)	62.03 (3.14)	61.89 (3.57)	61.90 (3.70)	62.80 (3.78)
All the meta-bases	RF chains	87.09 (2.61)	86.27 (2.06)	86.74 (1.86)	88.73 (2.02)	89.35 (2.03)	89.98 (1.69)

classifiers' errors in Table 17 were all lower than the MCE of 51.24% in both scenarios.

From Tables 15 and 16, we can see that the traditional categories of input meta-features, including simple, statistical and information-theoretical descriptors, alone produce results as good as those obtained with the new data characterization measures. The literature demonstrates that simple, statistical, and information-based input meta-features really have high discriminative power in metalearning, being widely used to describe several other previously extensively validated problems (Kalousis, 2002; Lemke et al., 2015; Parmezan et al., 2017). However, analyzing the average accuracy from the second scenario in Table 17, we conclude that the new descriptors and the traditional ones are beneficial for the problem in question. The main reason is: the input meta-features that most distinguish among classes tend to be different at each level of the classifier tree in Fig. 13. For example, MB4 is two levels ahead of MB9 in the classifier hierarchy of Fig. 13. Also, considering the results from Table 15, the simple characterization measures are the most discriminative in MB4 (Average rank = 2.90) as well as the graph-based descriptors do in MB9 (Average rank = 3.88). Although the relevance of one category of input meta-features over the others seems marginal from the Friedman rank point of view, symbolic metalearners that internally perform feature selection or classifier chains like those treated in Table 17 can recognize it (Parmezan et al., 2020).

7. Conclusion

Metalearning for feature selection algorithm recommendation consists of suggesting to the user one or more feature selection methods, from a set of candidate algorithms, for a new dataset. Such recommendation is driven by the past performance of the candidate methods in other datasets.

This paper addressed the problem of automatic choice of feature selection algorithms via metalearning by proposing a novel meta-feature engineering model that encompasses input meta-features and target meta-features. A set of new meta-features was also proposed in this study. We organized the input meta-features into eight categories: (i) simple, (ii) statistical, (iii) information-theoretical, (iv) complexity, (v) landmarking, (vi) based on symbolic models, (vii) based on images, and (viii) based on complex networks (graphs). The target meta-features included five individual performance indexes and one multi-criteria performance measure.

Our work made use of the proposed meta-feature engineering model to recommend feature selection algorithms based on information, distance, dependency, consistency, and precision measures. From 213 publicly available datasets, we developed a broad study building a meta-base with 1456 meta-examples described by 161 input meta-features and one target meta-feature. We decomposed the meta-base in terms of the target meta-feature into nine multiclass subproblems, as suggested in Parmezan et al. (2017). We then statistically analyzed the subproblems from three perspectives: (i) proportion between objects and classes; (ii) relevance of input meta-features for metalearning; and (iii) predictive performance of recommendation models for feature selection algorithms (meta-models).

An overview of the results indicated that the data characterization primarily via measures of simple, landmarking, image-based, and graph-based categories is promising for the problem of automatic choice of feature selection algorithms.

We found that the input meta-features from Average strength (weighted graph), Correlation degree centrality, Maximum local scan, Average local scan, Median local scan, Signal/noise ratio, Dispersion of the dataset, Error rate of the nearest neighbor classifier, and Ratio measure have higher discriminative power than the others. Thus, they are regarded as good candidates in the dataset representation for the task in question.

From a statistical analysis of the input meta-feature significance, we identified a minimum set of input meta-features (top-five input meta-features) that allowed us to discriminate, with high reliability and low computational cost, the performance of distinct feature selection algorithms.

We also demonstrated that it is possible to obtain good results in recommending feature selection algorithms, with hit rates of up to 90%, using the meta-feature engineering model proposed here. In practice, the higher the dimension ($N \times M$) of the test dataset(s) – dataset(s) for which we want to recommend one out of F feature selection algorithms –, the more efficient our approach will be when compared to exhaustive empirical assessments. To exemplify, if we adopt the minimum set of input meta-features, the computational cost of the recommender will be dominated by $O(V^3)$ — or $O(N^3)$ since V is equal to N. The cubic complexity time results from the combination of the asymptotic behavior of the 51 dataset characterization measures that make up the minimum set of input meta-features (Table 14), assuming that $N \gg M$ in the datasets for which we want to suggest algorithms. For scenarios where $M \gg N$, M ends up being decisive on such a cost. In any case, we can parallelize most descriptors' implementation to speed up their resolution. For example, the correlation degree centrality can be computed for each vertex separately but simultaneously. This design decision would reduce the impact of a cubic cost in practice. Most importantly, the recommendation is made without (i) human intervention or specialized knowledge and (ii) the need to apply F algorithms to each test dataset.

To the best of our knowledge, this paper is the first to evaluate such a broad and diverse meta-feature engineering model. Although we have designed the meta-features to assist in the feature selection algorithm recommendation task, they can be easily used in other metalearning scenarios, including classification, regression, or clustering algorithm selection for stationary (Ferrari & De Castro, 2015; Reif et al., 2014; Won Lee & Giraud-Carrier, 2013) and non-stationary (Anderson et al., 2019) problems. Our findings, coupled with a careful discussion of the strengths and weaknesses of the most prominent investigated meta-features, are an important contribution to automated machine learning.

Evaluating other metalearners to build suggestions, especially those that generate a ranking of labels as an output, is a subject of future work. Also, we plan to make available to the user a web tool that will integrate different meta-models in order to support the choice of feature selection algorithms automatically.

CRediT authorship contribution statement

Antonio Rafael Sabino Parmezan: Conceptualization, Methodology, Software, Investigation, Validation, Visualization, Writing – original draft. **Huei Diana Lee:** Conceptualization, Methodology, Validation, Writing – review & editing. **Newton Spolaôr:** Conceptualization, Methodology, Software, Investigation, Writing – original draft. **Feng Chung Wu:** Conceptualization, Methodology, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was financed in part by the Brazilian National Council for Scientific and Technological Development [grant numbers 140159/2017-7 and 142050/2019-9], and the Araucária Foundation for the Support of the Scientific and Technological Development of Paraná through a Research and Technological Productivity Scholarship for H. D. Lee [grant number 028/2019].

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2021.115589.

References

Aduviri, R., Matos, D., & Villanueva, E. (2018). Feature selection algorithm recommendation for gene expression data through gradient boosting and neural network metamodels. In *IEEE international conference on bioinformatics and biomedicine* (pp. 2726–2728).

Anderson, R., Koh, Y. S., Dobbie, G., & Bifet, A. (2019). Recurring concept meta-learning for evolving data streams. *Expert Systems with Applications*, *138*, Article 112832.

Arauzo-Azofra, A., Aznarte, J. L., & Benítez, J. M. (2011). Empirical study of feature selection methods based on individual feature evaluation for classification problems. *Expert Systems with Applications*, *38*(7), 8170–8177.

Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*, 245–271.

Bolón-Canedo, V., Sánchez-Maroño, N., & Alonso-Betanzos, A. (2013). A review of feature selection methods on synthetic data. *Knowledge and Information Systems*, *34*(3), 483–519.

Bolón-Canedo, V., Sánchez-Maroño, N., & Alonso-Betanzos, A. (2015). Recent advances and emerging challenges of feature selection in the context of big data. *Knowledge-Based Systems*, *86*, 33–45.

Breiman, L., Friedman, J., Stone, C., & Olshen, R. (1984). *The Wadsworth and Brooks-Cole statistics-probability series*, *Classification and regression trees*. California: Taylor & Francis.

Brezočnik, L., Fister, I., & Podgorelec, V. (2018). Swarm intelligence algorithms for feature selection: a review. *Applied Sciences*, *8*(9), 1521.

Brodley, C. E. (1993). Addressing the selective superiority problem: Automatic algorithm/model class selection. In *International conference on machine learning* (pp. 17–24).

Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, *40*(1), 16–28.

Chen, T., & Guestrin, C. (2016). Xgboost: a scalable tree boosting system. In *ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 785–794). ACM.

Costa, L. F., Rodrigues, F. A., Travieso, G., & Boas, P. R. V. (2007). Characterization of complex networks: a survey of measurements. *Advances in Physics*, *56*(1), 167–242.

Das, S. (2001). Filters, wrappers and a boosting-based hybrid for feature selection. In *International conference on machine learning* (pp. 74–81).

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis*, *1*(3), 131–156.

Dernoncourt, D., Hanczar, B., & Zucker, J.-D. (2014). Analysis of feature selection stability on high dimension and small sample data. *Computational Statistics & Data Analysis*, *71*, 681–693.

Dôres, S. N., Soares, C., & Ruiz, D. (2017). Effect of metalearning on feature selection employment. In *International workshop on automatic selection, configuration and composition of machine learning algorithms* (pp. 1–7).

Dy, J. G., & Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of Machine Learning Research*, *5*, 845–889.

Fayyad, U. M., & Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *International joint conference on artificial intelligence* (pp. 1022–1027). Morgan Kaufmann Publishers.

Ferrari, D. G., & De Castro, L. N. (2015). Clustering algorithm selection by meta-learning systems: a new distance-based problem characterization and ranking combination methods. *Information Sciences*, *301*, 181–194.

Filchenkov, A., & Pendryak, A. (2015). Datasets meta-feature description for recommending feature selection algorithm. *AINL-ISMW FRUCT*, 11–18.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, *7*(2), 179–188.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, *3*, 1289–1305.

Goswami, S., Chakrabarti, A., & Chakraborty, B. (2016). A proposal for recommendation of feature selection algorithm based on data set characteristics. *Journal of Universal Computer Science*, *22*(6), 760–781.

Grzymala-Busse, J. W., & Hu, M. (2000). A comparison of several approaches to missing attribute values in data mining. In *International conference on rough sets and current trends in computing* (pp. 378–385). Springer.

Han, J., Kamber, M., & Pei, J. (2011). *Data mining: concepts and techniques* (3rd ed.). California: Morgan Kaufmann.

Haralick, R. M., Shanmugam, K., & Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, *SMC-3*(6), 610–621.

John, G., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *International conference on machine learning* (pp. 167–173).

Kalousis, A. (2002). Algorithm selection via meta-learning (Ph.D. thesis), Centre Universiteire d'Informatique, Université de Genève.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1–2), 273–324.

Lee, H. D., Mendes, A. I., Spolaôr, N., Oliva, J. T., Parmezan, A. R. S., Wu, F. C., & Fonseca-Pinto, R. (2018). Dermoscopic assisted diagnosis in melanoma: Reviewing results, optimizing methodologies and quantifying empirical guidelines. *Knowledge-Based Systems*, *158*, 9–24.

Lee, J., & Olafsson, S. (2013). A meta-learning approach for determining the number of clusters with consideration of nearest neighbors. *Information Sciences*, *232*, 208–224.

Lemke, C., Budka, M., & Gabrys, B. (2015). Metalearning: a survey of trends and technologies. *Artificial Intelligence Review*, *44*(1), 117–130.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: a data perspective. *ACM Computing Surveys*, *50*(6), 94:1–94:45.

Liu, H., & Motoda, H. (2007). *Computational methods of feature selection*. Minnesota: Chapman & Hall/CRC Data Mining and Knowledge Discovery.

Mantovani, R. G., Rossi, A. L., Alcobaça, E., Vanschoren, J., & de Carvalho, A. C. (2019). A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves SVM classifiers. *Information Sciences*, *501*, 193–221.

Molina, L. C., Belanche, L., & Nebot, À. (2002). Feature selection algorithms: A survey and experimental evaluation. In *International conference on data mining* (pp. 306–313).

Montero-Manso, P., Athanasopoulos, G., Hyndman, R. J., & Talagala, T. S. (2020). FFORMA: feature-based forecast model averaging. *International Journal of Forecasting*, *36*(1), 86–92.

Parmezan, A. R. S., & Batista, G. E. A. P. A. (2015). A study of the use of complexity measures in the similarity search process adopted by kNN algorithm for time series prediction. In *IEEE international conference on machine learning and applications* (pp. 45–51). IEEE.

Parmezan, A. R. S., Lee, H. D., & Wu, F. C. (2017). Metalearning for choosing feature selection algorithms in data mining: Proposal of a new framework. *Expert Systems with Applications*, *75*, 1–24.

Parmezan, A. R. S., Silva, D. F., & Batista, G. E. A. P. A. (2020). A combination of local approaches for hierarchical music genre classification. In *International society for music information retrieval conference* (pp. 740–747). ISMIR.

Parmezan, A. R. S., Souza, V. M. A., & Batista, G. E. A. P. A. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Information Sciences*, *484*(5), 302–337.

Peng, Y., Flach, P. A., Soares, C., & Brazdil, P. (2002). Improved dataset characterisation for meta-learning. In *International conference on discovery science* (pp. 141–152).

Pfahringer, B., Bensusan, H., & Giraud-Carrier, C. G. (2000). Meta-learning by landmarking various learning algorithms. In *International conference on machine learning* (pp. 743–750).

Priebe, C. E., Conroy, J. M., Marchette, D. J., & Park, Y. (2005). Scan statistics on enron graphs. *Computational and Mathematical Organization Theory*, *11*(3), 229–247.

Reif, M., Shafait, F., Goldstein, M., Breuel, T., & Dengel, A. (2014). Automatic classifier selection for non-experts. *Pattern Analysis and Applications*, *17*(1), 83–96.

Rice, J. R. (1976). The algorithm selection problem. *Advances in Computers*, 65–118.

Sheikhpour, R., Sarram, M. A., Gharaghani, S., & Chahooki, M. A. Z. (2017). A survey on semi-supervised feature selection methods. *Pattern Recognition*, *64*, 141–158.

Shilbayeh, S., & Vadera, S. (2014). Feature selection in meta learning framework. In *Science and information conference* (pp. 269–275). IEEE.

Somol, P., & Novovicova, J. (2010). Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(11), 1921–1939.

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2009). Mining multi-label data. In *Data mining and knowledge discovery handbook* (pp. 667–685). Springer.

Wang, G., Song, Q., Sun, H., Zhang, X., Xu, B., & Zhou, Y. (2013). A feature subset selection algorithm automatic recommendation method. *Journal of Artificial Intelligence Research*, *47*, 1–34.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *The Morgan Kaufmann series in data management systems*, *Data mining: practical machine learning tools and techniques* (3rd ed.). Amsterdam: Morgan Kaufmann.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82.

Won Lee, J., & Giraud-Carrier, C. (2013). Automatic selection of classification learning algorithms for data mining practitioners. *Intelligent Data Analysis*, *17*, 665–678.

Yu, L., & Liu, H. (2003). Feature selection for high-dimensional data: a fast correlation-based filter solution. In *International conference on machine learning* (pp. 856–863).

Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, *5*, 1205–1224.