DE GRUYTER OPEN

Int. J. Appl. Math. Comput. Sci., 2017, Vol. 27, No. 4, 697–712
DOI: 10.1515/amcs-2017-0048

amcs

ON THE PREDICTIVE POWER OF META–FEATURES IN OPENML

BESIM BILALLI [a,*], ALBERTO ABELLÓ [a], TOMÀS ALUJA-BANET [b]

[a]Department of Service and Information System Engineering
Polytechnic University of Catalonia (BarcelonaTech), Campus Nord, Jordi Girona, 1–3, 08034, Barcelona, Spain
e-mail: {bbilalli,aabello}@essi.upc.edu

[b]Department of Statistics and Operations Research
Polytechnic University of Catalonia (BarcelonaTech), Campus Nord, Jordi Girona, 1–3, 08034, Barcelona, Spain
e-mail: tomas.aluja@upc.edu

The demand for performing data analysis is steadily rising. As a consequence, people of different profiles (i.e., non-experienced users) have started to analyze their data. However, this is challenging for them. A key step that poses difficulties and determines the success of the analysis is data mining (model/algorithm selection problem). Meta-learning is a technique used for assisting non-expert users in this step. The effectiveness of meta-learning is, however, largely dependent on the description/characterization of datasets (i.e., meta-features used for meta-learning). There is a need for improving the effectiveness of meta-learning by identifying and designing more predictive meta-features. In this work, we use a method from exploratory factor analysis to study the predictive power of different meta-features collected in OpenML, which is a collaborative machine learning platform that is designed to store and organize meta-data about datasets, data mining algorithms, models and their evaluations. We first use the method to extract latent features, which are abstract concepts that group together meta-features with common characteristics. Then, we study and visualize the relationship of the latent features with three different performance measures of four classification algorithms on hundreds of datasets available in OpenML, and we select the latent features with the highest predictive power. Finally, we use the selected latent features to perform meta-learning and we show that our method improves the meta-learning process. Furthermore, we design an easy to use application for retrieving different meta-data from OpenML as the biggest source of data in this domain.

Keywords: feature extraction, feature selection, meta-learning.

1. Introduction

Recently, more and more non-experts have been using data mining tools to perform data analysis. These users require off-the-shelf solutions that will assist them throughout the process. The process itself, also known as knowledge discovery, consists of several steps, such as *data selection*, *data pre-processing*, *data mining*, and *evaluation* or *interpretation* (Fayyad *et al.*, 1996); see Fig. 1.

The key step of the whole process is data mining. Yet the staggeringly large number of alternative algorithms that can be used makes this step challenging. Thus, non-experienced users become overwhelmed and require support (e.g., need to be recommended what data mining algorithm to use). Various techniques have emerged (Serban *et al.*, 2013) to provide that. Among them, one that has been the focus of research for long is meta-learning (Brazdil *et al.*, 2008; Bilalli *et al.*, 2017; Lemke *et al.*, 2015).

As we will show next, in short, meta-learning is a process that seeks to predict/find the performance of an algorithm on a given dataset. The ability of predicting the performance of different data mining algorithms allows one to rank the algorithms and therefore provide user support in data mining. However, the success of meta-learning depends on many factors. One of the most important factors is the set of *meta-features* used for meta-learning. Recall that there are two main ingredients in meta-learning: (i) the *dataset characteristics* or the *meta-features* plus the performance measure of algorithms on datasets or the *meta response*—these together define the *meta-data*, and (ii) the *meta-learner*. Yet, the primary

*Corresponding author

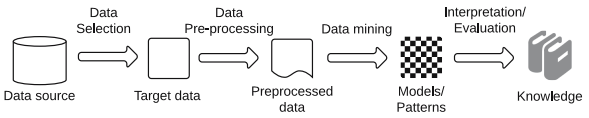

Fig. 1. Data analysis/knowledge discovery.

source of determining the success of meta-learning are the chosen *meta-features* (dataset characteristics).

In this paper, we provide a method for first extracting latent features, which basically group together meta-features with "common characteristics". Given the latent features, next we study their correlation to the performance measure that needs to be predicted. The reason to study the correlation at the latent feature level, rather than the meta-feature level, is that, first, there is no complete list of meta-features, so any analysis would lack completeness, and, second, even if there was such a list, the list would be so large that a sound analysis would not be feasible. Another possibility would be to study the relationship between a coarser group of meta-features and the different performance measures, and make generalizations out of that. For instance, Reif *et al*. (2014) study the relationship of the groups of meta-features (e.g., *statistical* meta-features) with some performance measures. We believe this kind of analysis hides the diversely significant relationships of different subsets of meta-features within the same coarse group (i.e., statistical meta-features). That is to say, some statistical features within the same group may have more significant relationships with the performance measure rather than some others in the same group. The analysis at this level overlooks this, and hence the generalizations that all statistical features behave similarly may be incorrect. That is why in this study we settle on a middle ground, where we neither study individual meta-features nor groups of such coarse granularity. We conduct the study at the latent feature level.

There have been many studies with regard to the use (what kinds of meta-data to be used) (Castiello *et al*., 2005) and selection (which are the most relevant) (Kalousis and Hilario, 2001) of meta-features or meta-data (Bilalli *et al*., 2016) in general. However, these studies have been performed independently and in specific domains. As a matter of fact, the amount of datasets and meta-data studied has been relatively small. With the appearance of OpenML (Vanschoren *et al*., 2014), however, the idea of collecting and generating meta-data and experiments has broadened. OpenML engages the whole machine learning community in the idea of collecting experiments, datasets and meta-data. As a matter of fact, the amount of available data and meta-data has naturally increased and is steadily increasing day by day. That is why our analysis is performed on data-sets and meta-data provided by OpenML as the biggest source of data for meta-learning. Specifically, as a first approach, our focus of analysis is classification problems.

Contributions. The main contributions of this paper can be summarized as follows:

- We use a traditional method of feature extraction and selection for a novel purpose of studying the predictive power of meta-features in a meta-learning scenario.
- We hand-craft the latent features behind the OpenML meta-features, then we study and visualize their predictive power for predicting different performance measures of different classification algorithms. As a novelty and in contrast to other works, we perform our analysis by splitting the datasets according to the meta-features that can be retrieved from them.
- We evaluate the effectiveness of the method for feature extraction and selection by performing meta-learning on top of OpenML data, and we show the obtained benefits.
- We develop a user-friendly tool that can be used (e.g., by data analysts) to generate meta-datasets (employed for meta-learning) out of OpenML.

The rest of the paper is organized as follows. In Section 2, an overview of meta-learning is given. The method for studying and visualizing the predictive power of meta-features is formally defined and explained in Section 3. In Section 4, we give an overview of OpenML and, more importantly, we report on results obtained after performing our method on top of OpenML. The related work is discussed in Section 5. Finally, Section 6 summarizes our work and outlines some future research.

2. Meta-learning

Most of the data analysis performed remains hidden and not reused. The vast amount of experience gathered from this analysis is not well exploited. The idea behind meta-learning is to exploit the knowledge gained out of

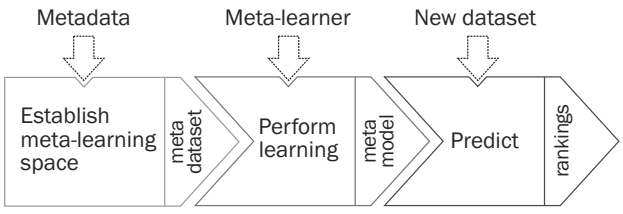

Fig. 2. Meta-learning process.

this experience. More precisely, meta-learning is the process of learning the relationships between datasets and data mining algorithms. Once different data-mining algorithms have been applied on different datasets, the idea is to use that knowledge when data mining algorithms need to be applied on new datasets. As depicted in Fig. 2, meta-learning consists of three steps.

First, a metalearning space is established using meta-data consisting of dataset characteristics (meta-features) and a performance measure (meta-response) for data mining algorithms on those particular datasets. Then, there comes the meta-learning phase. Here, a predictive meta-model is generated out of the meta-dataset constructed in the first phase. Finally, in the third step, when a new dataset comes, its characteristics are extracted and the predictive meta-model is used to predict the performance of a particular algorithm, for which the meta-model was built, on that dataset.

This technique can be used to rank different algorithms depending on their predicted performance on a new dataset. Hence, it can be employed to recommend data mining algorithms in the data mining step of the analysis.

The two main concepts of meta-learning are the meta-data and the meta-learner. In the following, we briefly discuss these two concepts.

2.1. Meta-data. Meta-data are the necessary information required to establish the meta-dataset. In our definition, they consist of (i) meta-features and (ii) a performance measure of the algorithm considered—meta-response. In statistics, the former are called *predictors* and the latter is called *response*.

Meta-features characterize a dataset, and initially the two following classes of measures have been proposed:

- *General:* include general information related to the dataset at hand. To a certain extent they are conceived to measure the complexity of the underlying problem. Some of them are the number of instances, the number of attributes, dataset dimensionality, the ratio of missing values, etc.
- *Statistical and information-theoretic:* describe attribute statistics and class distributions of a dataset sample. They include various summary statistics per attribute like mean, standard deviation, class entropy, etc.

Since the problem to be solved is usually a prediction problem, and a variable (or more) is defined to be the response, further meta-features measuring the association between the predictors and the response have been used. These measures are grouped into the *Landmarking and Model-Based* class (Pfahringer *et al.*, 2000; Peng *et al.*, 2002). This class is related to measures asserted with simple machine learning algorithms, the so-called *landmarkers*, and their derivatives based on the learned models. They include error rates and pairwise $1-p$ values obtained by landmarkers such as 1NN, DecisionStump or NaiveBayes. Yet, when performed on bigger datasets, these simple machine learning algorithms may yield significant computational cost.

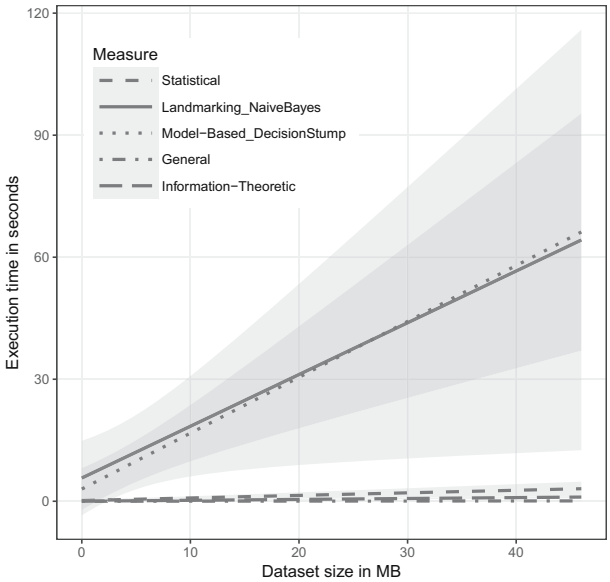

Fig. 3. Meta-feature extraction cost. Only the *Landmarking and Model-Based* class is represented through single measures. The rest of the classes are represented as the total execution times of their participating measures (see Table 2).

In order to assess the computational behaviour of the *Landmarking and Model-Based* class of measures, we performed an empirical analysis with 720 datasets. In addition to *Landmarking and Model-Based*, we calculated the performance of the other classes, too, and we show the comparison in terms of execution times in Fig. 3. Observe that only the *Landmarking and Model-Based* class is represented with individual measures such as NaiveBayes and DecisionStump. The other classes are represented as totals of their participating measures. For instance, the execution time of the *Statistical* class is calculated as the total execution time required to retrieve 24 individual measures like *Mean Standard Deviation*, *Mean Skewness*, *Mean Kurtosis*, etc. (see Table 2).

For the sake of presentation, instead of showing the scatter plot of the values of all the measures for each dataset, we show the fitted line (regression line) for each measure or class of measures. In addition, the grey areas around the lines denote the 95% interval of the prediction. One can immediately observe the steepness

Table 1. Classification algorithm performance measures.

Measure	Formula
Accuracy	$\frac{TP+TN}{TP+FP+FN+TN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
AUC	$\frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)$

TN: true negatives, TP: true positives,
FN: false negatives, FP: false positives.

of the slopes, and the wideness of the intervals around the lines representing *Landmarking and Model-Based* measures. The former indicates that with an increase in the dataset size, retrieving these measures becomes way more costly compared to the rest. The latter indicates that there is a high variability in the execution times for *Landmarking and Model-Based* measures. This means that, even if the size of a dataset is small, it can still be costly to retrieve these measures. This can happen for instance if the dataset contains a high number of features and a small number of instances. Hence, because of the way they are computed and their computational overhead, we do not consider *Landmarking and Model-Based* measures as classical dataset characteristics and they do not participate as meta-features in our experiments.

Performance measures (meta-response) are different outputs that can be obtained after the evaluation of data mining algorithms. Since we are dealing with classification problems, and so the algorithms we consider are of classification type, the performance is usually measured in terms of *predictive accuracy*, *precision*, *recall* or the *area under the ROC curve (AUC)*. In Table 1, formulas for calculating these measures are given. More precisely, classification algorithms are usually evaluated using 10-fold cross-validation (Kohavi, 1995).

2.2. Meta-learner. After having generated a meta-dataset with all the necessary meta-data, the goal is to build a predictive meta-model that will be able to predict the performance of an algorithm on a new dataset. Formally, the problem can be defined as follows. Given algorithm A and a limited number of training data $D = (\mathbf{x}_1, y_1) \dots (\mathbf{x}_n, y_n)$, the goal is to find a meta learner with good generalization performance. It is estimated by splitting D into disjoint training and validation sets $D^{(i)}_{\text{train}}$ and $D^{(i)}_{\text{valid}}$. Note that $\mathbf{x} \in x_1, x_2, \dots, x_n$ are the dataset characteristics and y_1 is a measure of the performance of algorithm A run on that particular dataset. Hence, $\mathbf{x}$ and y altogether

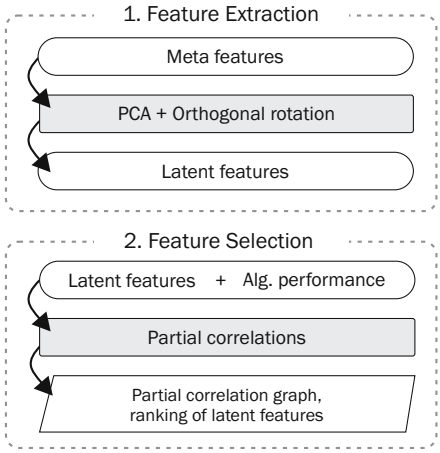

Fig. 4. Two stages of the method.

are the extracted meta-data. Various meta-learners have been used, such as *k-nearest neighbours*, *decision trees* and *support vector machines*, etc. (Bensusan and Giraud-Carrier, 2000a; Giraud-Carrier, 2005).

3. Predictive power of meta-features: Feature extraction and feature selection

As mentioned above, different and numerous meta-features can be used to define the meta-dataset for meta-learning. The list of meta-features may become very big, due to the *meta-meta* effect, referred to as meta^2 (Reif *et al.*, 2012). For instance, when a statistical characteristic needs to be taken over the continuous attributes of a given dataset, usually the *mean* of that statistic over the continuous attributes is taken. Let us say, if a general value for *skewness* is in question, commonly the *mean skewness* of all the continuous attributes is considered. However, this does not always need to be the case. One may opt for another statistic, different from the *mean*, for instance, the *skewness* of the *skewness* of continuous attributes, or even another, the *kurtosis* of the *skewness* of continuous attributes. As a result, the plethora of statistics that can be computed on top of other statistics may lead to an explosion of the number of meta-features that can be considered in a meta-learning process.

Therefore, the problem of choosing the most relevant and "minimal" (less computational overhead), set of meta-features is still present. Consequently, the chosen meta-features play a key role in determining the success of meta-learning.

In this section, we discuss the method that we use to study the relevance of the meta-features for predicting the performance measures of data mining algorithms. Our method consists of two steps, which are depicted in Figure 4.

In the first step, we perform a principal component analysis (PCA) and subsequently an orthogonal rotation on the complete set of meta-features that may be available. Even though this is a standard method in exploratory factor analysis, and has been used on different occasions (Morchid *et al.*, 2014), this is the first time that it is applied at the meta-feature level. Hence, the input required by this step is a meta-dataset consisting of dataset characteristics (meta-features). PCA followed by an orthogonal rotation allows us to **extract** the *latent features* and furthermore eliminate the redundant meta-features. Latent features are the abstract features that can be automatically extracted, yet they need to be manually interpreted. The automatic part of the process groups the meta-features with "common characteristics" into latent features. These common characteristics are the abstract concepts that need to be manually interpreted in order to define/describe the latent features.

After extracting the *latent features* we are able to perform the second step. The latent features are used as input for the second step; however, in addition, for all the datasets we compute a performance measure (e.g., predictive accuracy)—a *response feature* of an algorithm (i.e., the one we want to study). Hence, the input of the second step is a meta-dataset, but this time comprised of latent features of datasets and a performance measure of an algorithm run on the respective datasets.

In the second step, we perform a partial correlation analysis and generate a *partial correlation graph* that visualizes the relationship between *latent features* and the *response*. This allows us to **select** the *latent features* that are most relevant for predicting the *response*. Hence, at the end of the whole process we obtain a subset of *latent features*—expressed through meta-features—that can be next used for meta-learning. In the following, we briefly and formally introduce the two steps.

3.1. Principal component analysis. PCA (Hotelling, 1933) is the predominant linear dimensionality reduction technique, and it has been widely applied to datasets in all scientific domains, from social sciences and economics, through to biology and chemistry. In words, PCA seeks to reduce the dimension of a large number of directly observable features into a smaller set of indirectly observable ones—latent features. More precisely, the goals (Morchid *et al.*, 2014) of PCA are the following:

- extract the most important information from the dataset,
- compress the size of the dataset by keeping only this important information,
- explain and simplify the description of the dataset,
- analyze the structure of observations (instances) and variables.

In order to achieve these goals, PCA computes new features, which are called *principal components*. These are obtained as linear combinations of the original features. The first principle component is required to have the largest possible variance to "explain" the largest part of the variance of the dataset (i.e., meta-dataset). Then, the rest of the components are computed under the following constraints: (i) each component needs to be orthogonal to the previous one and (ii) each component needs to have the largest possible variance. The values of these new features are called *factor scores* and are geometrically interpreted as the *projections* of the instances onto the principal components. These are obtained from singular value decomposition (SVD) of the dataset X, with

$$X = P\Delta Q^T, \tag{1}$$

where P is an $m \times l$ matrix of left singular vectors, Q is an $n \times l$ matrix of right singular vectors and Δ is the diagonal matrix of singular values. Here l is the rank of the matrix X ($l \leq \min\{m, n\}$). The $m \times l$ matrix of factor scores denoted by F is obtained as $F = P\Delta$ and can be interpreted as a *projection* matrix, because multiplying X by Q gives the values of the *projections* of the observation on the principal components using Eqn. (1):

$$XQ = P\Delta Q^T Q = P\Delta = F. \tag{2}$$

Note that, in Eqn. (2), matrix F is generated using a standardized dataset (in our case a meta-dataset) while matrix X with $\dim(X) = (m, n)$, where m is the number of instances and n is the the number of features. PCA allows finding a subspace of size p, where the features are grouped depending on their projections onto the factor space. The feature groups actually form latent features/factors. A set of p factors $p \leq n$ is then selected. Each factor represents a certain part of the total variance of the dataset.

3.2. Orthogonal rotation. To facilitate interpretation, after having determined the number of components, the analysis usually involves a rotation of the components retained. Two types of rotations are mainly used: *orthogonal* (the new axes are required to be orthogonal to each other) and *oblique* (the new axes are not required to be orthogonal). Note that the part of variance explained by the total subspace after rotation is the same as before the rotation. In this paper, orthogonal rotation or, more precisely, the VARIMAX (Kaiser, 1958) method is chosen to perform a *transformation* of the original data. VARIMAX assumes that a simple solution means that each component has a small number of large loadings and a large number of zero loadings. Formally, it searches for a linear combination of the original factors such that the variance of the squared loadings is maximized, which

amounts to maximizing

$$v = \sum (q_{j,l}^2 - \bar{q}_l^2)^2,$$

with $q_{j,l}^2$ being the squared loading of the j-th variable of the matrix Q on the component l and $\bar{q}_l^2$ being the mean of the squared loadings. This rotation is performed using the diagonal matrix of singular values and the eigenvectors associated with the correlation matrix of X.

After the rotation, the set of factors (latent features) are more interpretable, and they, of course, are defined by their respective meta-features—the ones that are most correlated with them.

3.3. Partial correlation graphs. The first step produces a subset of candidate latent features for meta-learning. However, it does not provide a relevance measure of the latent features with respect to the response. In consequence, it does not necessarily retain only the latent features that are most relevant for predicting the response. These most relevant latent features with respect to the response out of the ones provided by the first step are derived here, in the second step. That is why, in this step, an additional feature (i.e., response) is attached to the derived set of latent features. The additional feature can be any of the performance measures of the algorithms evaluated over the datasets—instances of the meta-dataset. Given that, graphical models (i.e., partial correlation graphs) that represent the relationships between features can be generated.[1] The emphasis, of course, is on the relationships of the latent features and the response feature, rather than on the relationships between latent features. The latent features that have a very significant relationship with the response are the most relevant ones for predicting the response. Hence, we believe that retaining only the subset of latent features with very significant relationships with the response is sufficient for performing meta-learning. Furthermore, the set of relevant latent features may be different for different algorithms considered in a meta-learning framework. More formally, let $\mathbf{x}, \mathbf{y} \in R$ and $\mathbf{z}$ be a random vector. The partial correlation between $\mathbf{x}$ and $\mathbf{y}$, given $\mathbf{z}$, is a measure of association between $\mathbf{x}$ and $\mathbf{y}$ after removing the effect of $\mathbf{z}$. Specifically, $\rho(\mathbf{x}, \mathbf{y}|\mathbf{z})$ is the correlation between $\epsilon_{\mathbf{x}}$ and $\epsilon_{\mathbf{y}}$, where

$$\epsilon_{\mathbf{x}} = \mathbf{x} - \prod_{\mathbf{z}} \mathbf{x}, \quad \epsilon_{\mathbf{y}} = \mathbf{y} - \prod_{\mathbf{z}} \mathbf{y}.$$

Here, $\prod_{\mathbf{z}} \mathbf{x}$ is the projection of $\mathbf{x}$ onto the linear space spanned by $\mathbf{z}$. That is, $\prod_{\mathbf{z}} \mathbf{x} = \mathbf{z}\beta$, where β minimizes $\mathbb{E}[\mathbf{x} - \mathbf{z}\beta]^2$. In other words, $\prod_{\mathbf{z}} \mathbf{x}$ is the linear regression of $\mathbf{x}$ on $\mathbf{z}$. A similar argument applies to $\prod_{\mathbf{z}} \mathbf{y}$.

[1] Taking the partial correlations on latent features allows us to overcome the practical problem of the redundancy that may exist between the meta-features, since latent features are "orthogonal."

4. Experimental study on the predictive power of OpenML meta-features

In this section, we first give a brief description of OpenML. Then we discuss the meta-data it stores and the application we built to retrieve these meta-data in order to generate meta-datasets for meta-learning. After that, we describe the experimental setup which consists of applying the method for feature extraction and selection on OpenML meta-data, and we discuss the obtained results. Finally, we asses the performance of our method by meta-learning on 720 datasets, and we show the results obtained.

4.1. OpenML. It is an open science platform developed with the aim of allowing researchers to share their datasets, implementations and experiments (i.e., machine learning and data mining) so that they can easily be found and reused by others. It offers a web API through which new resources and results can be submitted, and has been integrated in a number of popular machine learning and data mining platforms, such as Weka, RapidMiner, KNIME, and data mining packages in R. They allow easy and automatic submission of new results.

4.1.1. Meta-data in OpenML. As previously mentioned, in our definition, meta-data consist of meta-features or, more precisely, dataset characteristics and a response feature or performance measures of different algorithms on datasets. In OpenML, for each uploaded dataset, 61 dataset characteristics (meta-features) are calculated. They are listed in Table 2. We classify each dataset characteristic into one of the following categories:

- *continuous*: if the dataset characteristic can be calculated only on datasets that contain continuous attributes,
- *categorical*: if the dataset characteristic can be calculated only on datasets that contain categorical attributes,
- *generic*: if the dataset characteristic can be calculated on any dataset.

Classifying meta-features into these three groups is very important, since, for instance, a *continuous* meta-feature cannot be calculated on datasets with only categorical attributes and vice versa, a *categorical* feature cannot be calculated on datasets with only continuous attributes. Hence, an analysis of meta-features should take this into account. To the best of our knowledge, however, no prior study considers grouping datasets according to the meta-features that can be extracted from them. Reif *et al*. (2014) recommend converting the *continuous* attributes into *categorical* (e.g., by discretization)

Table 2. Dataset characteristics in OpenML.

No	Name	Type	Class
1	Number of Numeric Attributes	*Continuous*	General
2	Percentage of Numeric Attributes	*Continuous*	General
3..6	Min[Means|Std|Kurtosis|Skewness] of Numeric Attributes	*Continuous*	Statistical
7..10	Mean[Means|Std|Kurtosis|Skewness] of Numeric Attributes	*Continuous*	Statistical
11..14	Max[Means|Std|Kurtosis|Skewness] of Numeric Attributes	*Continuous*	Statistical
15..17	Quartile [1|2|3] of Means of Numeric Attributes	*Continuous*	Statistical
18..20	Quartile [1|2|3] of Std of Numeric Attributes	*Continuous*	Statistical
21..23	Quartile [1|2|3] of Kurtosis of Numeric Attributes	*Continuous*	Statistical
24..26	Quartile [1|2|3] of Skewness of Numeric Attributes	*Continuous*	Statistical
27	Number of Categorical Attributes	*Categorical*	General
28	Number of Binary Attributes	*Categorical*	General
29	Percentage of Categorical Attributes	*Categorical*	General
30	Percentage of Binary Attributes	*Categorical*	General
31..33	[Min|Mean|Max] Attribute Entropy	*Categorical*	Information-Theoretic
34..36	Quartile [1|2|3] Attribute Entropy	*Categorical*	Information-Theoretic
37..39	[Min|Mean|Max] Mutual Information	*Categorical*	Information-Theoretic
40..42	Quartile [1|2|3] Mutual Information	*Categorical*	Information-Theoretic
43	Equivalent Number of Attributes	*Categorical*	Information-Theoretic
44	Noise to Signal Ratio	*Categorical*	Information-Theoretic
45..48	[Min|Mean|Max|Std] Attribute Distinct Values	*Categorical*	Statistical
49	Number of Instances	*Generic*	General
50	Number of Attributes	*Generic*	General
51	Dimensionality	*Generic*	General
52,53	[Number|Percentage] of Missing Values	*Generic*	General
54,55	[Number|Percentage] of Instances with Missing Values	*Generic*	General
56	Number of Classes	*Generic*	General
57	Class Entropy	*Generic*	Information-Theoretic
58,59	[Minority|Majority] Class Size	*Generic*	General
60,61	[Minority|Majority] Class Percentage	*Generic*	General

in order to be able to extract meta-features of *categorical* type in purely continuous datasets, otherwise the *categorical* features need to be replaced with missing values. However, applying such kinds of excessive transformations introduces noise. Thus, in our study (see Section 4.2.3), we group datasets based on their characteristics, and we perform our analysis on these groups separately. That is, the meta-features extracted and considered in accordance with the types are of attributes a dataset contains.

Regarding the performance measures of algorithms on datasets, all the performance measures defined in Table 1 are stored in OpenML for different classification algorithms. Hence, one can use the meta-data provided by OpenML in order to define meta-datasets for meta-learning.

4.1.2. Meta-data retrieval from OpenML. In order to create meta-datasets for meta-learning out of OpenML, one needs to have a good knowledge of the schema of the OpenML repository—the database consists of around 40 tables. This may pose challenges, especially to data analysts with a statistics background. In order to facilitate this process, we first developed a simple application, available for researchers[2], that is capable of generating a meta-dataset for any chosen data mining algorithm. A screenshot of the application is given in Fig. 5. After generating some meta-datasets we were able to continue with our studies, which is going to be explained next.

4.2. Experimental setup. In the following, we discuss our experimental setup which consists in applying the

[2]https://github.com/bbilalli/MetadataFromOpenML.

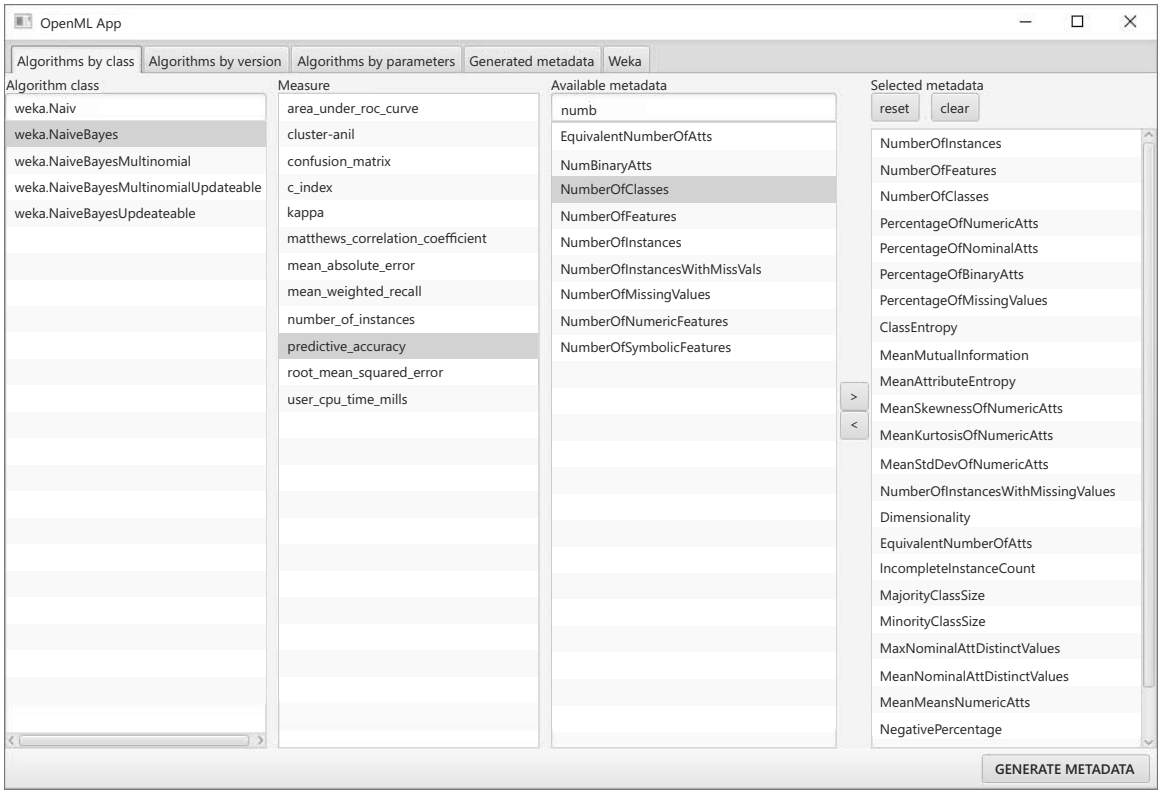

Fig. 5. Application for meta-data retrieval from OpenML.

method shown in Fig. 4 to meta-data retrieved from OpenML.

4.2.1. Retrieved meta-data. As previously mentioned, since our focus is on classification problems, we retrieved meta-data with regard to classification problems only. Hence, we retrieved meta-data for 720 datasets (classification problems)—very big datasets with respect to size in terms of features ($n > 1000$) or instances ($m > 1000000$) were not considered. The dataset characteristics (meta-features) and performance measures (meta-response) retrieved are listed in Tables 2 and 1, respectively. For the sake of experiments, the classification algorithms that we consider include *Decision Tree, Naive Bayes, JRip* and *Nearest Neighbor*.

4.2.2. PCA and orthogonal rotation for OpenML. As described in Eqn. (2), matrix F is generated using a standardized dataset matrix X with $\dim(X) = (m, n)$, where m is the number of observations ($m = 720$) and n is the number of meta-features ($n = 61$). A set of p factors (latent features) is then selected ($p = 14$). Each factor represents a certain part of the total variance of the meta-dataset. Several methods are used to estimate the correct number of factors to represent the data variance and the feature correlation. One that is commonly used is to retain all the factors that cumulatively represent a fair amount of the total variance (e.g., 80%). The number of factors needed to explain 80% of the total variance in this case is 14. These selected factors altogether represent 80.69% of the total variance. After determining the number of factors, in order to facilitate the interpretation and more clearly define the latent features, we perform a rotation (VARIMAX) of the retained factors (components). VARIMAX assumes that each factor has a small number of large loadings, and a large number of zero loadings.

Table 3 presents the most interesting factor loadings (i.e., with a threshold of ± 0.7) from matrix F for meta-features in the subspace represented by p first factors ($p = 14$). Factors are orthogonal and describe the correlation between the features in the original space representation. In the first column of Table 3, we show the hand-crafted latent concepts that stand behind each of the factors. They explain the meta-features shown in the second column. Finally, in the third column, factor loadings for the respective meta-features are shown. Factor loadings (correlations of meta-features with the latent factors), can range from -1 to 1. Loadings close to -1 or 1 indicate that the factor strongly affects the meta-feature. Loadings close to zero indicate that the factor has a weak effect on the meta-feature.

Table 3. Latent features obtained from PCA and VARIMAX.

Latent feature	Meta-feature	Corr.
Information of Categorical Attributes	MeanAttributeEntropy	0.853
	MinAttributeEntropy	0.925
	Q1AttributeEntropy	0.920
	Q2AttributeEntropy	0.878
Shape of Numeric Attributes	MeanKurtosisOfNumericAtts	0.933
	MeanSkewnessOfNumericAtts	0.874
	MinKurtosisOfNumericAtts	0.927
	Q1KurtosisOfNumericAtts	0.971
	Q1SkewnessOfNumericAtts	0.755
	Q2KurtosisOfNumericAtts	0.962
	Q2SkewnessOfNumericAtts	0.944
	Q3KurtosisOfNumericAtts	0.969
	Q3SkewnessOfNumericAtts	0.927
Variability of Numeric Attributes	MeanMeansOfNumericAtts	0.987
	MeanStdDevOfNumericAtts	-0.994
	MinMeansOfNumericAtts	0.999
	MaxStdDevOfNumericAtts	-0.999
Min. Variability of Num. Attributes	MinStdDevOfNumericAtts	0.821
	Q1MeansOfNumericAtts	0.719
	Qe1StdDevOfNumericAtts	0.887
Modality of Categorical Attributes	MaxNominalAttDistinctValues	0.899
	MeanNominalAttDistinctValues	0.874
	StdvNominalAttDistinctValues	0.942
Number of Instances	NumberOfInstances	0.978
	MajorityClassSize	0.935
	MinorityClassSize	0.843
Missing Values	#OfInstancesWithMissVals	0.816
	#OfMissingValues	0.866
	%OfInstancesWithMissVals	0.830
	%OfMissingValues	0.701
Dimensionality	NumberOfNumericFeatures	-0.955
	NumberOfFeatures	-0.956
	Dimensionality	-0.770
Information of the Response	ClassEntropy	-0.953
	NumberOfClasses	-0.773
	MajorityClassPercentage	0.770
Number of Cat. Atts.	NumberOfSymbolicFeatures	0.964
	NumberOfBinaryFeatures	0.966
Q3 Level of Num. Atts.	Q3MeansOfNumericAtts	0.974
	Q3StdDevOfNumericAtts	0.977
Shape of the Extreme Num. Attributes	MinSkewnessOfNumericAtts	0.719
	MaxKurtosisOfNumericAtts	-0.845
	MaxSkewnessOfNumericAtts	-0.794
Mutual Information	MeanMutualInformation	-0.942
	MinMutualInformation	-0.781
	MaxMutualInformation	-0.864
	Q1MutualInformation	-0.803
	Q2MutualInformation	-0.895
	Q3MutualInformation	-0.904
Noise to Signal	EquivalentNumberOfAtts	0.976
	NoiseToSignalRatio	0.975

4.2.3. Partial correlation analysis for OpenML meta-data. Having defined the latent features, it is time to rank them according to their relevance for predicting the performance of a given algorithm. In order to physically represent an abstract latent feature we use its corresponding meta-features. That is, a latent feature is physically represented as the average of the meta-features it explains (see Table 3). Taking the average of meta-features to define the latent concepts, and not their actual weights obtained in the specific instances of the datasets analyzed, gives a more robust measure of the latent feature, independent of the actual data, which can be easily generalized to future datasets by omitting non-existing meta-features or including new ones, provided that they are related with the actual concepts present in the latent features. We join the obtained latent features with the performance measures (i.e., accuracy, precision, recall and AUC) of the algorithms we consider for the corresponding datasets. Now, for each performance measure of every algorithm considered, partial correlation graphs can be generated in order to find the latent features that are more relevant for predicting the respective performance measures.

However, as previously mentioned, datasets retrieved from OpenML contain the same types of attributes, and hence no available meta-features can be calculated for all of them. Thus, we split the datasets into three sets:

- **combined**: consists of datasets that contain both continuous and categorical attributes (226 datasets)—all meta-features can be calculated (see Table 2);
- **continuous**: consists of datasets that do not contain categorical attributes (418 datasets)—only meta-features of *continuous* and *generic* type can be calculated (see Table 2);
- **categorical**: consists of datasets that do not contain numeric attributes (76 datasets)—only meta-features of *categorical* and *generic* type can be calculated (see Table 2).

As a consequence, partial correlation graphs are generated for all the performance measures of all the algorithms for every split of datasets separately. A study performed this way, distinguishing the groups of datasets, allows one to build a more customized meta-learning system. In addition, our hypothesis is that the datasets, the algorithms and, depending on the performance measures used, the relevance/importance of meta-features differ.

4.3. Experimental results. In this section, we first discuss the results obtained after applying the method for feature extraction and selection to four classification algorithms and three different performance measures. Next, we use the extracted/selected features for performing meta-learning and we show the obtained results.

4.3.1. Results on decision trees. In Fig. 6, the partial correlation graphs for decision trees are shown. Figure 6(a) shows the results with respect to Accuracy, 6(b) with respect to Precision and 6(c) with respect to AUC. Note that the graph for Recall is omitted since values identical with Accuracy are obtained (see weighted recall in Weka[3]).

In Fig. 6, shaded nodes represent the different types of meta- and latent features. The white nodes stand for the response features (performance measures). The presence of an edge from a latent feature node to a performance measure node indicates that there exists a significant correlation between the respective features. The correlation is considered to be significant, if the p-value of the correlation is $pval \leq 0.01$. The thickness of the edge represents the level of significance. Furthermore, dashed edges represent negative correlation and full edges stand for positive correlation. Finally, the different shades in the edges are used to denote the set of datasets where the significant correlation appears.

The nodes connected with short straight edges to the latent feature nodes represent the meta-features that define the respective latent features.

In Fig. 6(a), as can be observed, the latent feature that is most relevant for predicting accuracy is the *information of the response*, and it is attached with three edges. Thus, it is important in all the dataset splits we have considered. Furthermore, the edges are thick —the correlations are very significant, and the edges are full—the correlations are positive. This means that the higher the *information of the response* on a given dataset, the higher accuracy of decision tree applied of the on that dataset. The same effect can be observed for Precision (Fig. 6(b)), although the edges appear slightly thinner. Hence, for predicting Accuracy or Precision of decision trees, the *information of the response* is very important. The second most important latent feature which interestingly enough appears relevant for predicting all the measures is *mutual information*. However, the correlation appears very significant only in the Combined split and is positive. Another common feature for all the three measures is also *noise to signal*. It is negatively correlated and appears significant in the Combined split, meaning that the higher it is, the lower will be any of the measures considered.

The rest of the latent features appear less significant and are separately relevant for the given measures.

[3] `http://weka.sourceforge.net/doc.stable/weka/classifiers/Evaluation.html`.

Table 4. Most relevant latent features.

Split	Measure	Algorithm	Latent features: Generic: Info. of Response	Continuous: Min. Var. of Num. Atts.	Continuous: Dimensionality	Continuous: Shape of Extreme Num. Atts.	Categorical: Noise to Signal	Categorical: Info. of Cat. Atts.	Categorical: Mutual Info.
Combined	Accuracy	Decision Tree	3.5**e-22**				5.0**e-03**		7.1**e-12**
		Naive Bayes	3.6**e-21**						2.6**e-06**
		JRip	3.2**e-25**						8.2**e-13**
		Nearest Neighbor	5.8**e-17**				6.8**e-04**		2.8**e-09**
	Precision	Decision Tree	2.8**e-18**				6.1**e-03**		2.1**e-10**
		Naive Bayes	1.0**e-24**				5.7**e-03**		2.8**e-07**
		JRip	4.8**e-19**						1.2**e-09**
		Nearest Neighbor	1.1**e-17**				3.6**e-04**		6.9**e-10**
	AUC	Decision Tree					4.0**e-03**		4.8**e-06**
		Naive Bayes					4.2**e-05**		9.4**e-04**
		JRip					5.8**e-03**		2.1**e-04**
		Nearest Neighbor					5.4**e-05**		3.7**e-05**
Continuous	Accuracy	Decision Tree	2.6**e-30**	9.2**e-03**		9.7**e-03**			
		Naive Bayes	1.8**e-18**		1.6**e-03**				
		JRip	3.7**e-35**			9.5**e-03**			
		Nearest Neighbor	4.6**e-18**			6.9**e-03**			
	Precision	Decision Tree	3.7**e-23**	7.0**e-04**					
		Naive Bayes	9.8**e-22**		1.5**e-03**				
		JRip	2.5**e-28**			8.1**e-03**			
		Nearest Neighbor	5.1**e-16**			3.2**e-03**			
	AUC	Decision Tree				3.6**e-03**			
		Naive Bayes			4.3**e-03**				
		JRip				8.4**e-03**			
		Nearest Neighbor				2.8**e-03**			
Categorical	Accuracy	Decision Tree	2.0**e-04**						
		Naive Bayes	3.0**e-06**				2.7**e-03**	1.9**e-03**	
		JRip	5.4**e-05**					5.0**e-03**	
		Nearest Neighbor	8.7**e-04**				1.9**e-05**		
	Precision	Decision Tree	1.3**e-03**					7.4**e-03**	
		Naive Bayes	5.5**e-06**				8.0**e-03**	3.5**e-03**	8.2**e-03**
		JRip	1.4**e-03**						
		Nearest Neighbor	1.8**e-03**				2.3**e-05**		
	AUC	Decision Tree							
		Naive Bayes					1.7**e-04**		
		JRip							
		Nearest Neighbor					3.8**e-05**		

The columns highlighted in gray indicate the latent features with negative correlation.

4.3.2. Results on Naive Bayes, JRip and Nearest Neighbor. Taking into consideration the fact that all the algorithms we consider behave similarly (up to a certain degree) and given that for more algorithms, the graphs may not be very easy to follow, for the rest of algorithms we show the results in a concise table, namely, Table 4. For the sake of comparison we also add the results of decision tree.

We show the results in terms of *dataset splits*, *performance measures*, *algorithms* and *latent features*. Note that we omit the latent features that do not appear to be significant in any of the algorithms. Thus, the presence of a value for a latent feature denotes that a significant relationship exists between the latent feature and the corresponding performance measure for a given algorithm and dataset split. Furthermore, the value itself is the p-value of the correlation, which in the graphs was represented through the thickness of the edges. One

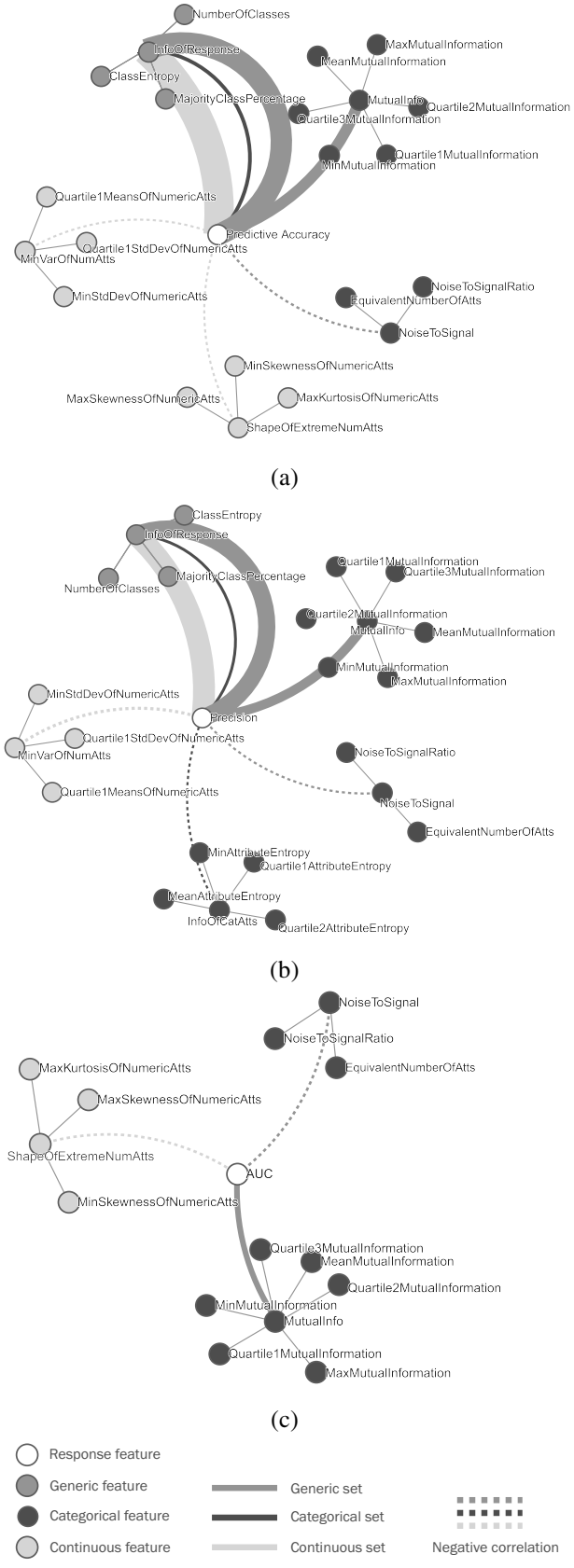

Fig. 6. Partial correlation graphs for decision trees: predictive accuracy (a), precision (b), AUC (c).

more thing to consider is the fact that the latent features highlighted in gray are negatively correlated, for all the rows in the table.

While reading the table, one can immediately observe some patterns. The first is that, independently of the split of datasets and independently of algorithms, Accuracy and Precision behave similarly. AUC, instead, depends on the split.

Other visible patterns are that *mutual information* is the measure that appears relevant for all the algorithms and for all the measures in the Combined split. On the other hand, *information of response* appears relevant in all the splits for all the algorithms for Accuracy and Precision. Furthermore, *shape of extreme numerical attributes*, *dimensionality* and *minimum variability of numeric attributes* appear significant in the Continuous split. This is due to the fact that these latent features are defined only for *continuous* meta-features. On the other hand, *noise to signal*, *information of categorical attributes* and *mutual information* appear relevant in the Categorical split. This is so because they are defined only for *categorical* features. Finally, an interesting fact about the Categorical split is that, for the AUC measure, in Decision Tree and JRip, no latent features appear relevant and for the rest of the algorithms only one latent feature appears to be such. In addition to this, observing the significance of the correlations of the AUC measure in all the splits, we can realize that they are usually less significant compared to the other measures. The former and the latter, altogether, indicate that AUC may be more difficult to predict, in comparison to the other measures.

4.3.3. Results on meta-learning. The method for studying the predictive power of meta-features allows us to define latent features and then, by measuring their relevance for predicting a performance measure, it helps select the most relevant ones. Therefore, at the end, only a subset of features is retained. In terms of meta-learning, using a subset of features instead of the complete set (i.e., 61) has many benefits. First, it saves computational effort when meta-features need to be retrieved. Secondly, fewer features means less computational time when building models (i.e., meta-models). Finally, since the retained features are latent features, the models built on top of them are more interpretable. Yet, in order to enjoy these benefits, the models built with the selected features need to perform well. That is, they need to be as good as, or even better than, the models built using all the meta-features. In order to check the latter, we trained a regression model (i.e., meta-model) on top of 720 datasets using *Random Forest* as a meta-learner. We evaluated the performance of the regression model using leave-one-out cross-validation. For measuring the performance of the meta-learner we used the root mean squared error (RMSE), which is often employed as a

Table 5. RMSE values of the predicted measures for all the splits.

Split	Measure	Alg.	RMSE: Feature extraction/selection	RMSE: All features
Combined	Accuracy	DT	**0.097**	0.123
		NB	**0.109**	0.129
		JRip	**0.095**	0.121
		kNN	**0.115**	0.140
	Precision	DT	**0.109**	0.130
		NB	**0.099**	0.125
		JRip	**0.108**	0.129
		kNN	**0.115**	0.138
	AUC	DT	**0.125**	**0.125**
		NB	**0.098**	0.104
		JRip	**0.120**	0.124
		kNN	**0.109**	0.119
Continuous	Accuracy	DT	**0.091**	0.102
		NB	**0.108**	0.122
		JRip	**0.090**	0.103
		kNN	**0.118**	0.130
	Precision	DT	**0.100**	0.108
		NB	**0.099**	0.114
		JRip	**0.094**	0.102
		kNN	**0.112**	0.123
	AUC	DT	**0.106**	0.107
		NB	**0.103**	0.106
		JRip	**0.110**	0.118
		kNN	**0.123**	0.127
Categorical	Accuracy	DT	0.138	**0.136**
		NB	**0.119**	0.128
		JRip	0.132	**0.131**
		kNN	0.151	**0.144**
	Precision	DT	0.153	**0.148**
		NB	**0.126**	**0.126**
		JRip	0.150	**0.145**
		kNN	0.155	**0.149**
	AUC	DT	0.168	**0.160**
		NB	**0.115**	0.123
		JRip	0.164	**0.156**
		kNN	0.181	**0.165**

The bold numbers indicate the best results per row.

measure of precision and can also serve as a confidence indicator for the predictions.

In Table 5, the column *Feature extraction/selection* shows the RMSE values of the trained regression models for the sets of features suggested (the union of the relevant latent features per split is taken) and the column *All features* shows the RMSE values for the complete set of meta-features. Furthermore, the results are classified in terms of *splits of the datasets*, *measures* and *classification algorithms* used. It can be observed that meta-learning with the feature extraction/selection method applied performs better than when all the features are used. The only exceptions are in the Categorical split. We believe this is due to that fact that latent features that appear relevant in the Categorical split are less significant compared with the ones in the rest of the splits—they are of the order of 10^{-4} (see Table 4). This furthermore is due to the fact that the Categorical split consists of only 76 datasets—significance values are affected by the sample size.

Table 6. RMSE values for the Categorical split.

Split	Measure	Alg.	RMSE: Feature extraction/selection	RMSE: Feature extraction	RMSE: All features
Categorical	Accuracy	DT	0.138	**0.131**	0.136
		JRip	0.132	**0.126**	0.131
		kNN	0.151	**0.139**	0.144
	Precision	DT	0.153	**0.147**	0.148
		JRip	0.150	**0.141**	0.145
		kNN	0.155	**0.142**	0.149
	AUC	DT	0.168	**0.154**	0.160
		JRip	0.164	**0.149**	0.156
		kNN	0.181	**0.165**	**0.165**

The bold numbers indicate the best results per row.

Note that we can think of significance as a measure of confidence. That is, the more significant the correlation, the more confident you can be about the predictive power of the feature under consideration. Hence, since the correlations in the Categorical split are less significant, we cannot be confident that the selected features are exactly the ones with more predictive power. In order to remedy this problem, we repeated the evaluation for the problematic cases of the Categorical split, although, this time using all the extracted latent features. The results are shown in the column *Feature extraction* in Table 6; we keep the previous results, too, for the sake of comparison. It can be observed that using all the extracted latent features the results improve, and they even become better than when using all the meta-features. This indicates that the extracted latent features are more robust than the original meta-features. It also suggest that when faced with small sample sizes (i.e., small number of datasets), we can opt for using only the first step of the method depicted in Fig. 4 (i.e., only *Feature extraction*).

5. Related work

Meta-feature definition. Most of the focus with regard to the meta-data in meta-learning has been on defining various dataset characteristics that can be used in meta-learning. The first attempt to characterize datasets was done by Rendell *et al*. (1987). Yet the description of a dataset in terms of its information/statistical measures for the first time appears within the framework of the STATLOG project (Michie *et al*., 1994). Fifteen dataset characteristics were used. This set of characteristics was later employed in various studies for solving the algorithm selection problem (Brazdil *et al*., 1994; Todorovski and Dzeroski, 1999). Sohn (1999) notices that some of the characteristics are highly correlated, and she omits the redundant ones in her study. Castiello *et al*. (2005) provide formulas for different data characteristics (meta-features) and theoretically discuss their relevance. However, their assumptions are based on intuition and theoretical knowledge. As a consequence, the conclusions are more generic.

An alternative approach to characterize datasets called landmarking has been proposed by Pfahringer *et al*. (2000) as well as Bensusan and Giraud-Carrier (2000b). An intuitive idea behind landmarking is that the performance of a simple learner, called a landmarker, can be used to predict the performance of given candidate algorithms. Landmarking measures have been evaluated and have shown to perform well in many works (Fürnkranz and Petrak, 2001; Bensusan and Kalousis, 2001; Pfahringer *et al*., 2000; Reif *et al*., 2014). The usefulness of these measures comes with a price though, which is the computational cost. Yet, none of the studies, apart from that by Bensusan and Kalousis (2001), properly acknowledge this fact. This is mainly because the studies are performed on a small number of datasets and on the ones of small sizes. For instance, Reif *et al*. (2014) perform studies on top of 54 datasets. Another group of measures, quite related to landmarking, includes model-based measures. The idea is to create a model from the data and use its properties as feature values. The model used in this context is typically a decision tree (Peng *et al*., 2002; Bensusan *et al*., 2000). These measures have been evaluated by Peng *et al*. (2002) and Reif *et al*. (2014). However, similarly to landmarking, they induce computational overhead when computed on datasets of bigger sizes.

Meta-feature selection. The first attempt at meta-feature selection appeared in the meta-learning framework of zooming-ranking (Todorovski *et al*., 2000). In this study, some experiments are shown where a classical feature selection method is applied to select relevant features. Kalousis and Hilario (2001) study the meta-feature selection problem, too. However, their method is constrained to finding relevant features for pairs of algorithms. This is because their definition of meta-learning is based on detecting the best classification algorithm in a context of pairs of algorithms. In the work of Reif *et al*. (2014), an empirical evaluation of different categories of meta-features in the context of their suitability for predicting classification accuracies for a number of standard classifiers can be found. In addition, an automatic feature selection method is applied to the complete set of meta-features used. However, no finer details of the feature selection method are given. Furthermore, the number of datasets used is very small.

6. Conclusions and future work

In this work, we used a method to tackle the problem of meta-feature extraction and selection. It relies on a rigorous mathematical framework and is beneficial for improving the success of meta-learning tasks. It consists in first extracting latent features out of meta-features, and then, by studying and visualizing the relationships of the latent features with the response (i.e., the performance measures of algorithms), it allows selecting the most relevant or informative latent features.

After applying the method to data retrieved from OpenML, we were able to observe that (i) no latent features are similarly relevant for predicting the performance of different classification algorithms, and vice versa—for predicting different performance measures for the same classification algorithms, (ii) no latent features are similarly relevant when meta-learning space consists of datasets with specific types of attributes—splitting the datasets in accordance to the meta-features that can be extracted from them was a novelty compared with previous works, and it played a decisive role in our analysis. For example, the latent features relevant in a set of datasets with only continuous attributes are not the same as the latent features relevant in a set of datasets with only categorical attributes, (iii) the method for meta-feature extraction/selection improves the meta-learning process.

Having observed this, we claim that meta-feature extraction/selection is a necessary pre-processing step for meta-learning. Moreover, we agree that the meta-learning space needs to be specifically customized taking into consideration the available datasets, algorithms and performance measures that need to be predicted in a meta-learning framework.

Our main future work consists in incorporating the method of meta-feature extraction/selection into the meta-learning framework we have been developing. This will allow us to extend the application of our method to other classification algorithms. Furthermore, we plan to expand our studies by also considering regression problems, which outnumber the classification problems in OpenML.

Acknowledgment

This research has been funded by the European Commission through the Erasmus Mundus Joint Doctorate *Information Technologies for Business Intelligence—Doctoral College* (IT4BI-DC).

References

Bensusan, H. and Giraud-Carrier, C. (2000a). Casa batló is in passeig de grácia or how landmark performances can describe tasks, *Proceedings of the ECML-00 Workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination, Barcelona, Spain*, pp. 29–46.

Bensusan, H. and Giraud-Carrier, C.G. (2000b). Discovering task neighbourhoods through landmark learning performances, *Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Lyon, France*, pp. 325–330.

Bensusan, H., Giraud-Carrier, C.G. and Kennedy, C.J. (2000). A higher-order approach to meta-learning, *Proceedings of the International Conference on Inductive Logic Programming, London, UK*.

Bensusan, H. and Kalousis, A. (2001). Estimating the predictive accuracy of a classifier, *in* M. Bramer (Ed.), *Principles of Data Mining. Undergraduate Topics in Computer Science*, Springer, London, pp. 25–36.

Bilalli, B., Abelló, A., Aluja-Banet, T. and Wrembel, R. (2016). Towards intelligent data analysis: The metadata challenge, *Proceedings of the International Conference on Internet of Things and Big Data, Rome, Italy*, pp. 331–338.

Bilalli, B., Abelló, A., Aluja-Banet, T. and Wrembel, R. (2017). Intelligent assistance for data pre-processing, *Computer Standards & Interfaces*, DOI: 10.1016/j.csi.2017.05.004, (in press).

Brazdil, P., Gama, J.A. and Henery, B. (1994). Characterizing the applicability of classification algorithms using meta-level learning, *Proceedings of the European Conference on Machine Learning, Catania, Italy*, pp. 83–102.

Brazdil, P., Giraud-Carrier, C., Soares, C. and Vilalta, R. (2008). *Metalearning: Applications to Data Mining*, 1st Edn., Springer Publishing Company, Berlin.

Castiello, C., Castellano, G. and Fanelli, A.M. (2005). Meta-data: Characterization of input features for meta-learning, *Proceedings of the International Conference on Modeling Decisions for Artificial Intelligence, Tsukuba, Japan*, pp. 457–468.

Fayyad, U.M., Piatetsky-Shapiro, G. and Smyth, P. (1996). From data mining to knowledge discovery in databases, *AI Magazine* **17**(3): 1–34.

Fürnkranz, J. and Petrak, J. (2001). An evaluation of landmarking variants, *Proceedings of the ECML/PKDD Workshop on Integrating Aspects of Data Mining, Decision Support and Meta-Learning, Freiburg, Germany*, pp. 57–68.

Giraud-Carrier, C. (2005). The data mining advisor: Meta-learning at the service of practitioners, *Proceedings of the International Conference on Machine Learning and Applications, Los Angeles, CA, USA*, pp. 113–119.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *Journal of Educational Psychology* **24**(6): 417–441.

Kaiser, H.F. (1958). The varimax criterion for analytic rotation in factor analysis, *Psychometrika* **23**(3): 187–200.

Kalousis, A. and Hilario, M. (2001). Feature selection for meta-learning, *Proceedings of the International Conference on Knowledge Discovery and Data Mining, Hong Kong, China*, pp. 222–233.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection, *Proceedings of the International Joint Conference on Artificial Intelligence, Montréal, Québec, Canada*, pp. 1137–1143.

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

Michie, D., Spiegelhalter, D.J., Taylor, C.C. and Campbell, J. (Eds.) (1994). *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, Chichester.

Morchid, M., Dufour, R., Bousquet, P., Linarès, G. and Torres-Moreno, J. (2014). Feature selection using principal component analysis for massive retweet detection, *Pattern Recognition Letters* **49**: 33–39.

Peng, Y., Flach, P.A., Soares, C. and Brazdil, P. (2002). Improved dataset characterisation for meta-learning, *Proceedings of the 5th International Conference on Discovery Science, Lübeck, Germany,* pp. 141–152.

Pfahringer, B., Bensusan, H. and Giraud-Carrier, C.G. (2000). Meta-learning by landmarking various learning algorithms, *Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, USA*, pp. 743–750.

Reif, M., Shafait, F. and Dengel, A. (2012). Meta2-features: Providing meta-learners more information, *35th German Conference on Artificial Intelligence, Staarbrücken, Germany*.

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

Rendell, L., Seshu, R. and Tcheng, D. (1987). Layered concept-learning and dynamically-variable bias management, *Proceedings of the International Joint Conference on Artificial Intelligence, Milan, Italy*, pp. 308–314.

Serban, F., Vanschoren, J., Kietz, J. and Bernstein, A. (2013). A survey of intelligent assistants for data analysis, *ACM Computing Surveys* **45**(3): 31:1–31:35.

Sohn, S.Y. (1999). Meta analysis of classification algorithms for pattern recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**(11): 1137–1144.

Todorovski, L., Brazdil, P. and Soares, C. (2000). Report on the experiments with feature selection in meta-level learning, *Proceedings of the PKDD Workshop on Data Mining, Lyon, France*, pp. 27–39.

Todorovski, L. and Dzeroski, S. (1999). Experiments in meta-level learning with ILP, *Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Prague, Czech Republic*, pp. 98–106.

Vanschoren, J., van Rijn, J.N., Bischl, B. and Torgo, L. (2014). OpenML: Networked science in machine learning, *ACM SIGKDD Explorations Newsletter* **15**(2): 49–60.

Besim Bilalli received his Master's degree in computer engineering from the Sapienza University of Rome in 2011. He is currently working towards his PhD degree with the GESSI Research Group, Universitat Politécnica de Catalunya (UPC), within the IT4BI-DC Erasmus Mundus Joint Doctorate. His research interests lie in the areas of data mining, data pre-processing and meta-learning.

Alberto Abelló obtained his doctorate at UPC in 2002 and has worked as a lecturer and researcher at that university since then. Currently he is an associate professor. He has held research stays at the University of Granada (Spain), Technische Universität Darmstadt (Germany), University Claude Bernard Lyon 1 (France) and the University of the Republic (Uruguay). He has participated in more than 10 national research projects or networks of excellence, and has signed R&D agreements with companies such as Hewlett Packard, Zurich Insurance, SAP or the World Health Organization. Currently he is a local coordinator of the IT4BI-DC Erasmus Mundus PhD programme.

Tomàs Aluja-Banet, a professor of statistics and data analysis of UPC since 1983, a former director of the Department of Statistics and Operations Research, a vice-dean for statistics in the Mathematics and Statistics Faculty and a vice-dean for corporate relations of the Barcelona School of Informatics. At present he is responsible for MIRI (data science master) at UPC. His research interests include methodological aspects of multivariate analysis, segmentation trees, data fusion, as well as PLS path models with their software implementation.

Received: 29 November 2016
Revised: 11 April 2017
Accepted: 8 August 2017