S3Mining: A model-driven engineering approach for supporting novice data miners in selecting suitable classifiers

Roberto Espinosa[a], Diego García-Saiz[*,b], Marta Zorrilla[b], José Jacobo Zubcoff[c], Jose-Norberto Mazón[d]

[a] *Wake Research Group, Universidad Tecnológica de Chile INACAP, Chile*
[b] *Dpto. de Ingeniería Informática y Electrónica, Universidad de Cantabria, Santander, Spain*
[c] *Wake Research Group, Dpto. Ciencias del Mar y Biología Aplicada, Universidad de Alicante, Spain*
[d] *Wake Research Group, Dpto. Lenguajes y Sistemas Informáticos, Instituto Universitario de Investigación Informática, Universidad de Alicante, Spain*

Keywords:
Data mining
Knowledge base
Model-driven engineering
Meta-learning
Novice data miners
Model-driven

A B S T R A C T

Data mining has proven to be very useful in order to extract information from data in many different contexts. However, due to the complexity of data mining techniques, it is required the know-how of an expert in this field to select and use them. Actually, adequately applying data mining is out of the reach of novice users which have expertise in their area of work, but lack skills to employ these techniques. In this paper, we use both model-driven engineering and scientific workflow standards and tools in order to develop named S3Mining framework, which supports novice users in the process of selecting the data mining classification algorithm that better fits with their data and goal. To this aim, this selection process uses the past experiences of expert data miners with the application of classification techniques over their own datasets. The contributions of our S3Mining framework are as follows: (i) an approach to create a knowledge base which stores the past experiences of experts users, (ii) a process that provides the expert users with utilities for the construction of classifiers' recommenders based on the existing knowledge base, (iii) a system that allows novice data miners to use these recommenders for discovering the classifiers that better fit for solving their problem at hand, and (iv) a public implementation of the framework's workflows. Finally, an experimental evaluation has been conducted to shown the feasibility of our framework.

1. Introduction

In recent years, the profusely use of information and communication technologies has led to an exponential growth of generated data. Data come from everywhere: social networks, educational platforms such as e-Learning Management Systems, sensors within Internet of the Things (IoT), open data from the public sector, among others. These are enough evidence to state that this "datification" process [1] leads to a "big data" world. Consequently, data is currently recognized as an essential asset useful for gaining insights and supporting decision making process in many domains. More and more professionals with data analysis skills are therefore needed, the currently known as data scientists [2]. These professionals require mastering techniques and technologies to extract knowledge from data, such as data mining. Data mining and knowledge discovery is defined as the process of applying data analysis and discovery algorithms to find knowledge patterns over a collection of data [3]. The application of data mining techniques has been tested to be successful in different fields such as educational context [4–6], economics [7], energy supply sector [8] or IoT [9].

Data mining is an intrinsically complex process [10,11] that requires, among other tasks, selecting the most suitable data mining algorithm for a given dataset. Due to the fact that there is no single algorithm that performs best on every dataset, as stated by the "No Free Lunch" theorem [12], many experimentation and expertise is required for identifying the best algorithm depending on the features of each dataset, so novice data miners can be overwhelmed. Consequently, obtaining reliable and useful knowledge from data mining requires the know-how of an expert in order to determine what data mining tech-

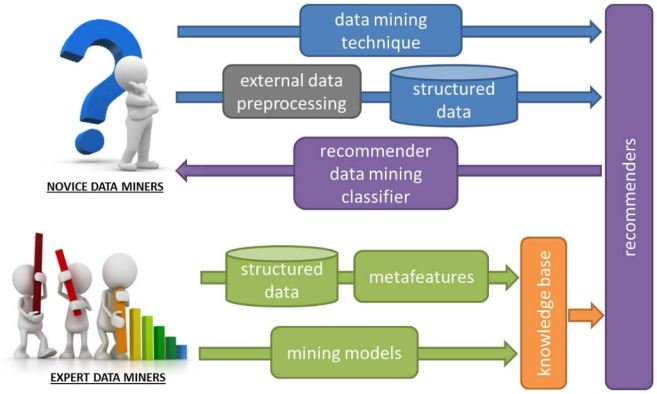

Fig. 1. Overview of our framework.

niques and parameters-setting are appropriate for being applied to the data sources according to user's requirements and the features of the data.

In this paper, we consider two different kinds of data miners, i.e., expert data miners and novice data miners:

Expert data miners. They are professionals that master the process of applying data analysis algorithms to find knowledge patterns over a collection of data [3]. To do so, they have wide expertise on data-preprocessing, data management, and advance statistics.
Novice data miners. They are professionals who have expertise in their work domain but without full algorithmic skills to make the most of data mining. These novice data miners require novel approaches that support them in selecting what algorithm is better to apply on their input datasets [13,14].

Currently, there exist many tools addressed to perform the data mining process but very few are focused to support novice data miners to assess data mining algorithms and select the best algorithm for a dataset at hand. To fill this gap, this paper defines a framework named S3Mining, which stands for "Supporting Selection of Suitable data Mining algorithms". It is based on the meta-learning concept: applying data mining algorithms on meta-data (meta-features) extracted from previous experiments in order to better understand the behavior of such algorithms and know which are the most suitable in solving different kinds of problems [15], i.e., meta-learning searches for the correlations between meta-data and the performance reached by the algorithms.

Our framework relies on the collaborative spirit of the open science initiative. According to [16], open science can be defined as "transparent and accessible knowledge that is shared and developed through collaborative networks". Open science thus implies research data sharing and stewardship to automate the generation of knowledge bases in different fields of research [17]. Interestingly, a recent study [18] argues that researchers are willing to share data due to a variety of factors, such as (i) regulative pressure by journals and normative pressure at a discipline level; and (ii) perceived career benefit and scholarly altruism at an individual level. Therefore, our framework is a convenient open science tool that allows expert users contribute with their knowledge and experiments, while novice users learn and develop their skills in the data mining field by applying these techniques on their datasets.

An overview of our framework is shown in Fig. 1. Expert data miners launch experiments on mining models with the aim of storing their results along with meta-features extracted from the input structured data in a knowledge base (KB). Of course, this KB can be incrementally enriched from executing mining experiments performed by expert users over time. This KB provides information for building data mining recommenders which enable novice data miners to obtain an algorithm or a ranking of suitable algorithms, considering the type of data mining technique they are interested in and given the problem domain to be solved.[1] It is worth noting that data pre-processing techniques are required to handle outliers, missing values, and data transformations. Expert data miners can pre-process data by themselves, but novice data miners need support. We would like to point out that dealing in detail with data pre-processing techniques is out of the scope of this paper, thus assuming that datasets are already pre-preprocessed by an expert user. However, we would like to highlight some interesting approaches that could be incorporated in our framework for data-preprocessing (labeled as "external data pre-processing" in Fig. 1). For example, the approach presented in [19] performs data pre-processing in an automatic manner with the support of meta-learning, while an approach for pipelining methods to facilitate further automated data pre-processing in presented in [20]. Also, there are domain-oriented data preprocessing approaches for characterizing automated data pre-processing such us the approach presented in [21] for environmental data.

Therefore, our framework has three main workflows: (i) expert data miners perform mining experiments to populate a knowledge base containing meta-features and classification models, (ii) expert data miners build an algorithm recommender that uses all the required information from the knowledge base, and (iii) novice data miners use this recommender to apply the suggested algorithm over their own dataset.

Such a framework must be formally defined at the same time that uniformity of information that is manipulated must be kept. Also, this framework should be independent from a specific database management system that manages all information with the aim of being easily reused. Likewise, all the process of extracting meta-features from data mining experiments, building a recommender and supporting its use should be formalized. To this end, we have developed our S3Mining framework by using some well-known standards and tools coming from (i) model-driven software engineering [22,23], and (ii) scientific workflows [24,25].

This paper is therefore a step forward to support selection of classification algorithms for novice data miners, making it easy to learn and acquire experience in data mining by using both model-driven engineering and scientific workflow standards and tools. Specifically, there are four main contributions that complement our previous work that focused on non-expert users [26]:

1. An approach to create a knowledge base that collects all the information about what an expert data miner considers relevant for applying data mining algorithms to data sources.
2. A process that provides the expert user with utilities for the construction of data mining recommenders based on the existing knowledge base.
3. A system that allows novice data miners to transparently use the recommenders in order to discover the classification algorithms that better fit for solving their problem at hand.
4. A public implementation of the workflows in Taverna,[2] which are available on the Web.[3]

Finally, we test the flexibility and feasibility of our proposal in two experiments. The first one is focused on educational data mining [6,27] (application of data mining in the educational context), an area of research with a great relevance due to the growing use of e-learning platforms in all education levels. The second one is addressed to general purpose data mining, and it has been conducted with datasets coming from UCI repository.[4]

[1] In its current state, our framework has workflows for recommending classification data mining algorithms but, of course, it could be extended so the data contained in the KB can be used for making recommendations of other type of data mining techniques.
[2] http://www.taverna.org.uk/.
[3] http://www.myexperiment.org/users/workflows/16696.
[4] https://archive.ics.uci.edu/ml/.

The remainder of this paper is structured as follows: the related work is addressed in Section 2. In Section 3 the different elements of our framework are introduced, along with a detailed description of them, while the conducted experiments are described in Section 4. Finally, conclusions and future work are sketched in Section 5.

2. Related work

This section summarizes some of the most important works related with our approach. Despite Knowledge Discovery for Databases (KDD) has grown immensely and attracted more focus from both research and industry in the last years, few tools have been developed in order to assist data miners in their job. Current data mining tools allow users to manually build KDD workflows and select each step from a large pool of possible solutions but they do not guide user to decide which algorithm is the best for a certain purpose, for instance. Furthermore, if we join this shortage with the recent high demand of data scientists, the proposal and implementation of intelligent data assistants (IDA) oriented to this goal must be investigated.

User-friendly data mining has emerged as a challenge in recent bibliography related to support novice data miners. Some user-friendly approaches are focused on providing interactive systems, an adaptive and effective communication between human users and computer systems [28] where the user is guided through the data mining process. FIU-Miner [29] is an integrated system that facilitates users to conduct ad hoc data mining tasks. It provides a user-friendly GUI to allow users to rapidly configure their experiments. Dimitropoulos et al. [30] proposed another scalable, user-friendly, and interactive data mining platform, designed for analyzing large heterogeneous data sets. The main drawback of these systems is that they require expertise on KDD process, since they focus on easing the application of different techniques in each step of the process. Therefore, novice users with little knowledge of data mining could not take advantage of these approaches. Additionally, there are some proposals that are intended to assist novice users in applying data mining focused on some specific application domain. For example, in [31], an alternative is proposed to generate genomic sequences, allowing the extraction of gene features from an annotation file while controlling for several quality filters and maintaining a user friendly graphical environment. In [32], a specific user-friendly tool for applying data mining in biology is proposed. Specifically, authors provide biologists with tools to investigate the associations between genes and encoded proteins. In the educational field, some of the authors of this work, developed ElWM [33], a web tool with the aim of helping instructors involved in distance education to discover their students' behavior profiles and models about how they navigate and work in their virtual courses offered in Learning Content Management Systems, such as Blackboard or Moodle. An extended version of this tool is described in [34].

There are other proposals that have addressed the issue of assisting users in business intelligence tasks. The so-called "self-service" business intelligence aims to enable non-expert users to make well-informed decisions when required, by letting them navigate "situational data" in a "surf and save" mode [35] i.e., data that have a narrow focus on a specific business problem and, typically, a short lifespan for a small group of users. This solution is focused on "On Line Analytical Processing" (OLAP), and more advanced data analysis techniques (such as data mining) are overcome.

Also, it is worth noting that there are several data mining ontologies that could be used as a basis of intelligent techniques for applying data mining algorithms. For example, OntoDM [36] is a top-level ontology for data mining concepts that describes basic entities aimed at covering the whole data-mining domain, while EXPO ontology [37] is focused on modeling scientific experiments. A more complete ontology is DMOP [38] which not only describes learning algorithms (including their internal mechanisms and models), but also workflows. Furthermore, a large set of data mining operators are described in the KD ontology [39] and the eProPlan ontology [40]. Regarding data mining workflows, the KDDONTO ontology [41] aims at both discovering suitable KD algorithms and describing workflows of KDD processes. It is mainly focused on concepts related to inputs and outputs of the algorithms and any pre and post-conditions for their use. Also, the Ontology-Based Meta-Mining of Knowledge Discovery Workflows [42] is aimed at supporting workflow construction for the knowledge discovery process. Moreover, in [43] authors propose a specific ontology to describe machine learning experiments in a standardized manner for supporting a collaborative approach to the analysis of learning algorithms (further developed in [44]). There are some projects that allow scientific community to contribute with their experimentation in improving the knowledge discovery process. The Machine Learning Experiment Database developed by University of Leuven [45] offers a Web tool to store the experiments performed in a database and query it. The e-LICO project funded by the Seventh Framework Programme [46] developed a knowledge-driven data mining assistant which relies on a data mining ontology to plan the mining process and propose ranked workflows for a given application problem [42]. Regarding what meta-features to use, in general, measurable properties of data sets and algorithms are chosen, for instance, statistical or information-theoretical measures [47], landmarkers [48] or model properties such as the average ratio of bias, variance error, or their sensitivity to noise [49] among others. The data context and its complexity for learning task are also used [50].

Also, ontologies are used in approaches that guide data miners in their work. For example, Charest and Delisle [51] propose an ontology to create an intelligent data mining assistant in order not to only provide novice data miners with a tool for data interpretation, but also to manage the inherent complexities associated with effectively using the available plethora of data mining tools, methods and algorithms.

Intelligent support for data miners is also encouraged in [52] where authors state that even students at the end of a long-term data mining course (i.e., novice data miners) were overwhelmed by typical data mining task. To overcome this scenario, they propose an intelligent data mining assistant by using semantic web technologies.

Finally, in [19], authors propose an intelligent approach based on metalearning in order to guide novice users in data preprocessing before performing data analysis. Authors analyze a wide range of data pre-processing techniques and a set of classification algorithms in order to automatically suggest the transformations that improve the quality of the results of the algorithm on the dataset. While this approach is focused on using metalearning for preparation of data for analysis, the starting point of our approach is different, since it is based on using the meta-features of data sources to support novice data miner in selecting suitable algorithms.

3. Supporting novice data miners in selecting suitable algorithms

Our S3Mining framework (Supporting novice data miners in Selecting Suitable mining algorithms) is composed of three main processes that use computer standards and tools coming from model-driven engineering and scientific workflows. They aim to guide expert data miners in generating some assets that later help novice data miners to apply suitable algorithms for specific goals. The main processes (and their assets) within our S3Mining framework are as follows:

1. Creating and populating a knowledge base to automatically keep know-how of expert data miners by storing meta-features extracted from conducted data mining projects.
2. Supporting expert data miners to build data mining algorithms' recommenders by using the previously created knowledge base via experiments.
3. Guiding novice data miners in the execution of the aforementioned recommenders in order to get suitable algorithms to be applied to their input dataset.

Every process in our framework is designed on the basis of model-driven software development in order to (i) easily collect meta-features from data mining experiments and represent them into the knowledge base, thus keeping uniformity with the information that is manipulated, (ii) manage all information independently from a specific database management system, thus achieving a platform-independent framework, and (iii) use model transformations to automatically generate our recommenders from the knowledge base.

According to [53], standardization of data mining does not mean to formally describe what data mining is or what it does, but instead encouraging using standards that support the data mining process. In this paper, different model-driven software development standards and techniques have been used. For instance, a metamodel has been specified to be able to create models that represent a knowledge base containing meta-features from data mining experiments, and a set of model-driven transformations have been developed to get these meta-features from experiments and to derive and generate the knowledge base as well as building the recommenders. Importantly, the processes of our S3Mining framework are defined by means of scientific workflow models in order to orchestrate these model-driven transformations together with execution of the data mining experiments. The development of these scientific workflows and their use within a model-driven development approach allow us to formalize the creation of the knowledge base from a meta-learning perspective (i.e., including information about the behavior of different data mining algorithms with regard of the metadata of the sources) and their use to generate data mining algorithms' recommenders.

Our S3Mining framework is implemented by using different tools: (i) scientific workflows have been implemented in the Taverna Workflow Tool,[5] (ii) metamodels and transformations have been implemented in Eclipse Modeling Framework (EMF),[6] and (iii) a RESTFul layer has been developed to create a set of web services to be used by Taverna scientific workflows to execute EMF artifacts. Finally, it is worth noting that all scientific workflows have been published within MyExperiment platform,[7] thus allowing community to reuse our approach.

It is worth noting that, although our framework is conceived for any data mining technique, in this paper we only focus on data mining classification tasks.

3.1. Creating and populating the knowledge base

The process that supports expert data miners to add meta-features from data mining experiments in the knowledge base is shown in Fig. 2. The process begins when expert data miners introduce input structured data sources together with their mining requirements: selection of mining algorithms and their settings, the attribute to predict, the meta-features to compute, and metric and method for performance evaluation. Next, a scientific workflow is used to compute meta-features of the input data sources chosen by the expert (e.g., domain of data, performance measure, etc.). Also, results of executing the corresponding data mining algorithms on these input structured data sources are acquired. All this information is stored in our knowledge base through the definition of models conformed to a specific metamodel on data mining experiments. As we pointed out in Section 1, we recall that input datasets must be previously preprocessing by the expert.

This process is specified as a scientific workflow (see Fig. 3) that allows expert data miners to (i) create a knowledge base in which mining models from experiments are stored, (ii) properly configure mining parameters, and (iii) process the input structured datasets as required. This workflow has been designed by using MyExperiment and

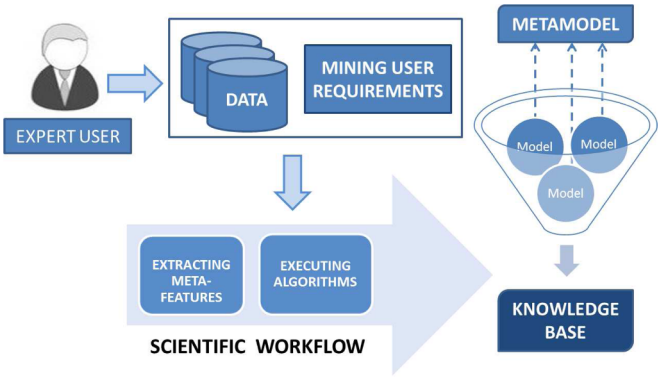

Fig. 2. Developing and feeding our knowledge base.

it can be accessed on the Web.[8]

3.1.1. Process set up

The workflow begins with the selection of input structured datasets. Then, expert users according to their criteria and expertise will launch the configuration parameter phase in order to set up the following mining parameters:

1. Data mining task that the expert user wants to apply (i.e., classification, regression, clustering, etc.). Selection of this data mining task is done in the *Select_DataMining_Task* box.[9]
2. Performance metrics to evaluate the mining models (e.g., accuracy, f-measure, sensitivity, specificity etc.). Expert user sets this performance metrics to be measured in the *Select_Measure_of_Performance* box.[10]
3. Method for measuring the performance of the mining classifier (e.g., cross validation, hold-out, leave-one-out, etc.). This selection can be made and configured in the *Select_Method_of_Performance* box.

Once the configuration parameter is done, the *REST_Get_Files_Information* box extracts meta-feature values of the input dataset (i.e., number of attributes, number of instances, numerical and nominal attributes percentages, etc.). These meta-features, chosen by the expert, are later stored in the knowledge base.

3.1.2. Mining models

Next step consists on execution of the subworkflow for the application of mining algorithms according to the parameter setting introduced by the expert (see *DataMiningAlgorithm_NestedWorkflow* in Fig. 3). Datasets, data mining algorithms, measure of performance, and the method for performance evaluation are the inputs of this subworkflow, while models and their performance metrics given by application of data mining algorithms are the output. The *REST_DataMining_Algorithms* box aims at using some mining library (for example, libraries from well-known tools as Weka[11]).

3.1.3. Extraction of meta-features

In the subworkflow *Metafeatures_NestedWorkflow*, different types of meta-features are considered and computed:

- Simple or general meta-features, such as the number of attributes, the number of instances, the type of attributes (numerical,

[5] http://www.taverna.org.uk/.
[6] https://eclipse.org/modeling/emf/.
[7] http://www.myexperiment.org/users/workflows/16696.
[8] http://www.myexperiment.org/workflows/3843.html.
[9] We would like to recall that experiments conducted in this paper focuses on classification.
[10] Although we use accuracy and f-measure in our experiments in Section 4, other metrics can be defined here and used.
[11] http://community.pentaho.com/projects/data-mining/.

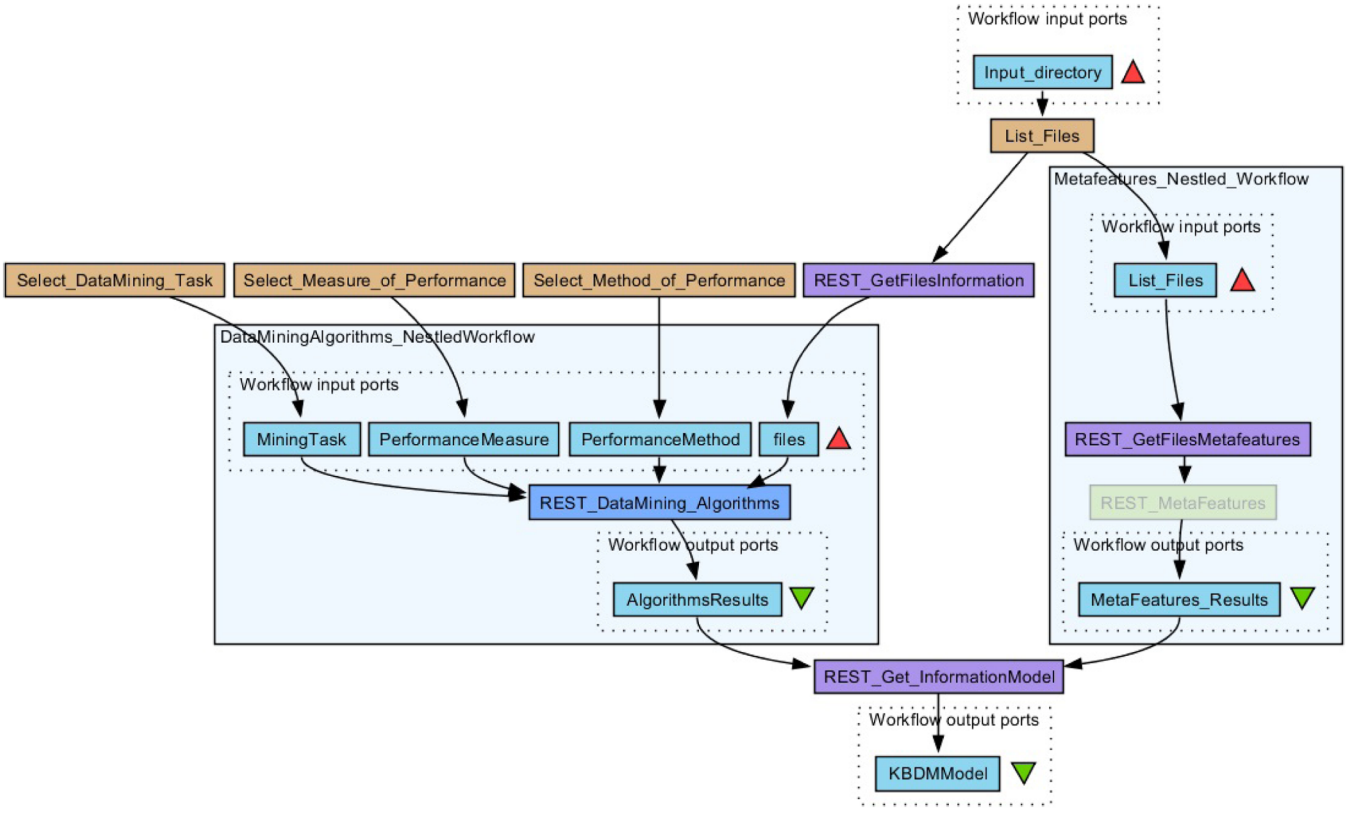

Fig. 3. Scientific workflow for creating the knowledge base.

categorical or mixed), the number of values of the target attribute and dimensionality of the dataset, i.e., the ratio between the number of attributes and the number of instances.

- Statistical meta-features (e.e., skew, kurtosis among others) to measure the distribution of attributes and their correlation [54,55].
- Information theoretic features used for characterizing datasets containing categorical attributes such as class entropy or noise to signal ratio [49].
- Model-based meta-features, which collect the structural shape and size of a decision tree trained on the datasets [56].
- Landmarkers, which are meta-features calculated as the performance measures achieved by using simple classifiers [48].
- Contextual features, i.e., characteristics related to dataset domain [50].

We also consider extraction of meta-features in the sense of some recent works [50,57] that have used a set of data complexity metrics (provided by DCoL tool [58] which measures characteristics of the data independently of the learning method) as meta-features. A description of the DCoL meta-features is included in Appendix B.

3.1.4. Knowledge base as a set of models

Our knowledge base aims to represent in a structured and homogeneous manner all the meta-features previously extracted, as well as the results of the mining models. Following the model-driven paradigm, our knowledge base is uniform and automatically created as a repository of models, which conforms to a metamodel that gathers the previous described meta-features from data mining experiments that experts perform. This Data Mining Knowledge Base metamodel is named as *DMKB*.

The elements that compose our *DMKB* are not restricted to certain meta-features, since the *DMKB* metamodel supports creating new features as required. The definition of our metamodel (see Figs. 4 and 5) is based on an analysis of several data mining ontologies (see Section 2).

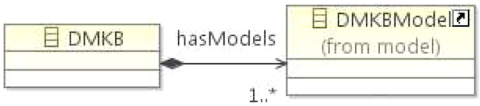

Fig. 4. DKMB metamodel that considers a knowledge base set of models.

In order to facilitate the future manipulation of information stored in the knowledge base, the *DMKB* metamodel is designed to group together all the mining information, but also allowing that the information of each data source can be stored as individual models, according to the *DMKBModel* metamodel (see Fig. 4). Then, the models that conform to the overall *DMKB* metamodel can be considered as a collection of all individual models (see Figs. 4 and 5). We refer reader to Appendix A for a description of each class that composes the aforementioned metamodels.

3.1.5. Transformations supporting knowledge base creation

In this section, we introduce how our knowledge base is created conforming *DMKB* metamodel. Specifically, a text-to-model transformation is developed in order to create a *DMKBModel*. Transformation is defined by using the set of classes generated by EMF to dynamically create different elements that conforms the *DMKB* metamodel. The Knowledge Base is created by means of a *Factory* interface and every element from the input data sets and data mining results are added with their corresponding information.

3.2. Building the data mining algorithms recommender

Fig. 6 shows the process to be executed by the expert data miner in order to configure those parameters required to build data mining algorithms' recommenders. The expert user must first build an input data file by querying the knowledge base to ask for any meta-feature previously stored, thus filtering useful instances for the recommenders.

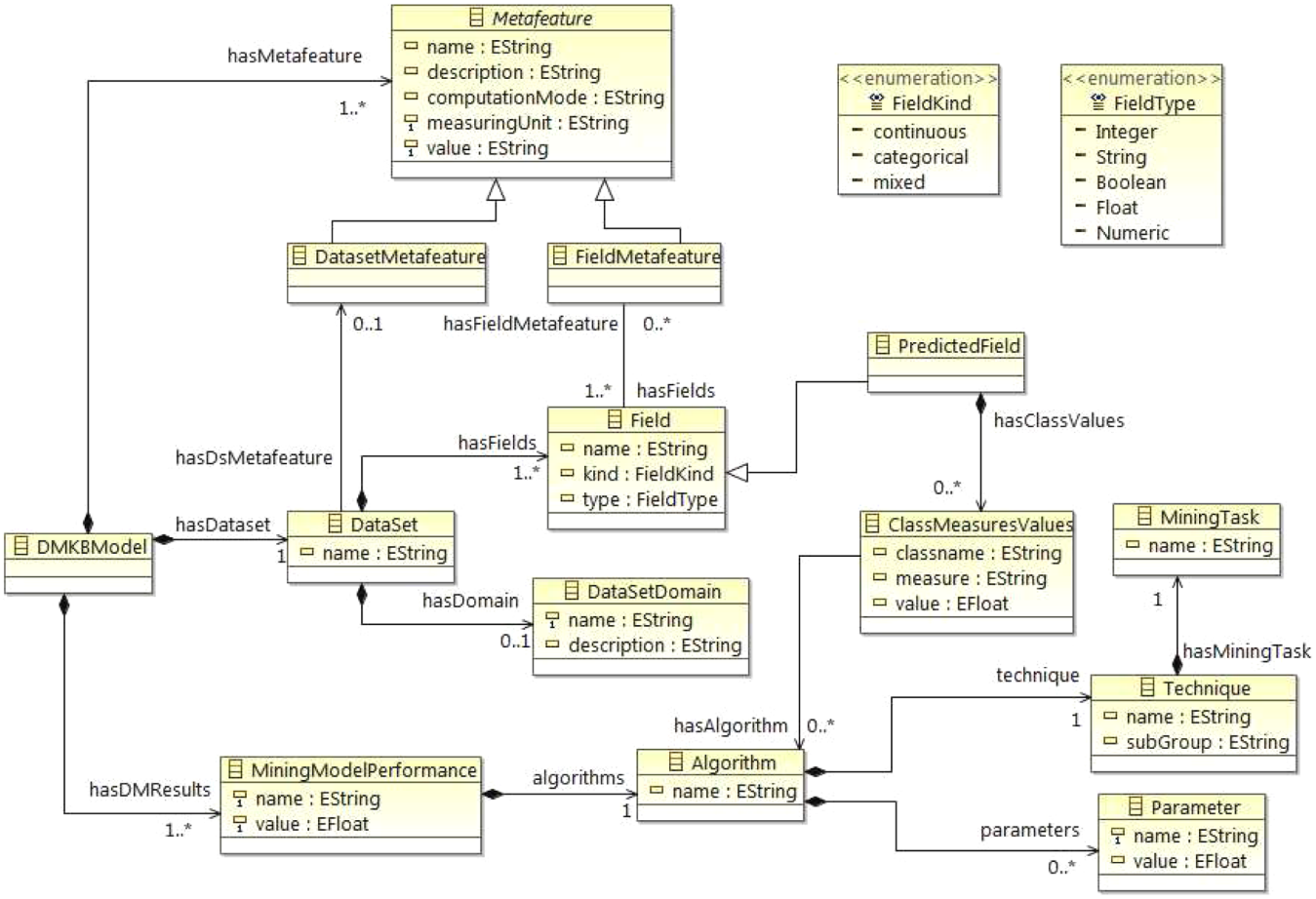

Fig. 5. Metamodel for building models that belong to our knowledge base.

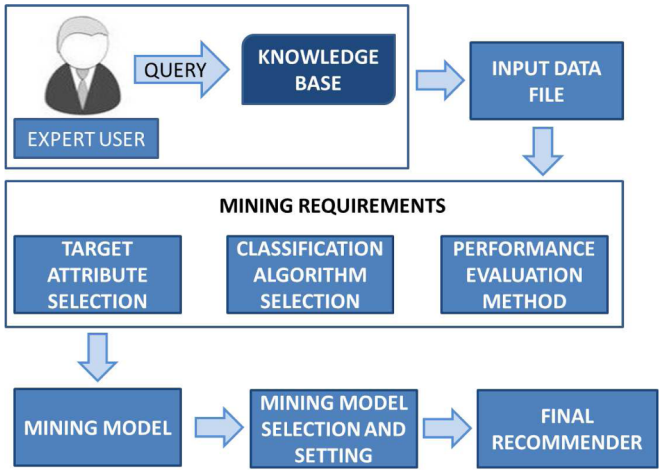

Fig. 6. Building our data mining recommenders.

Then, expert must configure several parameters: the target attribute, the approach to built the recommenders, and the performance evaluation method to be used (i.e., accuracy or f-measure, among others). Finally, when expert data miner runs the selected approach on the input data file, the recommenders are obtained. Different approaches can be used for building the recommenders, e.g., in this paper we focus on generating two kinds of recommenders (that later help novice users in selecting classifiers for their new incoming datasets), namely (i) recommenders with meta-classifiers, which select the best expected classifier, and (ii) recommenders with meta-regressor, which generates a ranking of classifiers ordered by the chosen performance metric.

3.2.1. *Scientific workflow for recommender construction*

A scientific workflow has been created for the expert data miner by using MyExperiment[12] to be able to set parameters and decide which are the best recommenders (see Fig. 7).

For each recommender, the expert data miner can configure each one of the following parameters: select the data mining profile (*Select_DataMining_Profile* box), the data mining measure to evaluate the performance of the recommender (*Select_DataMining_Measure*), the testing validation method (*Select_Testing_Validation_Method*), and the mining algorithm to be executed (*Select_Recommender_Algorithm*). The information for the recommender to properly works is provided by the knowledge base, acquired by the *REST_GetDMKB* box. In the *REST_Recommender* box, the recommender is created. Also, the process to compare the obtained result with previous ones is performed. Finally, the result is shown in the *DomainRecommender* box (domain refers to the fact that the recommender can be feed with both open-domain data or specific-domain data). Additionally, the obtained *.arff* files are given to the expert user too (*arff_file* box).

3.2.2. *Transformation for generating a recommender from the knowledge base*

This section discusses the model-to-text transformation for obtaining the required information from the knowledge base (represented as models, according to the metamodel presented in Fig. 4). Specifically, the transformation creates text files, with valid *.arff* format,

[12] http://www.myexperiment.org/workflows/4522.html.

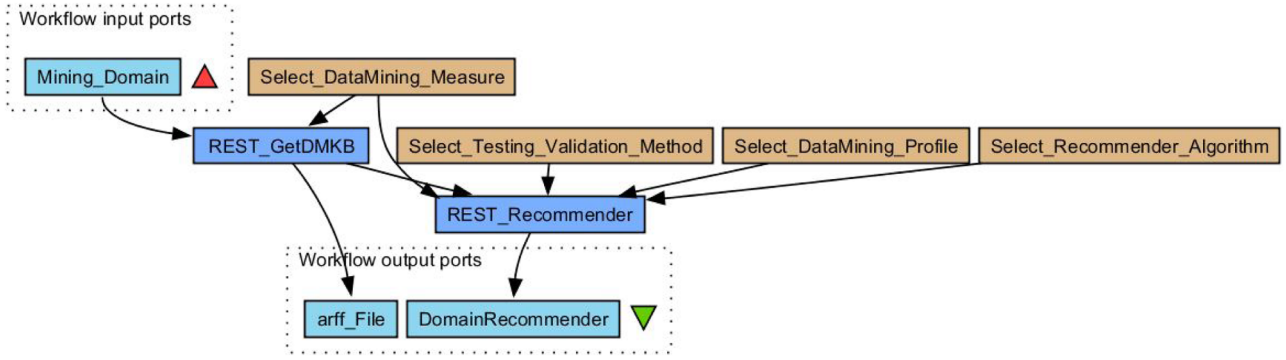

Fig. 7. Scientific workflow for recommenders construction.

containing the meta-features from the models belonging to the knowledge base. Our recommender system is then built from the information stored in these *.arff* files. Acceleo[13] is used to transform the selected *DMKB* models to an *.arff* file. Acceleo is an open-source code generator to be used within EMF for creating transformations that derive text from models. Our transformation will take those models that forms the knowledge base as input, while a valid *.arff* file is derived as output.

Acceleo works by creating a template where excerpts of code access the input model elements and transform these elements into the corresponding output code (i.e., text). A *.arff* file structure has two sections. The first section is the header information, which is followed by the instances, i.e. data to be processed. The header of the *.arff* file contains the name of the relation, a list of the attributes, and their types. This transformation is executed for each of the selected algorithms by the expert, thus obtaining one *.arff* file for each algorithm. An example of such a *.arff* file is shown in Code 1.

3.3. Using the recommender

Fig. 8 shows how a novice data miner may transparently use a recommender and get a ranking of suitable algorithms. Expected inputs to be provided by novice data miner are: (i) requirements, i.e. kind of mining problem to solve, and (ii) input datasets. Then, meta-features from the datasets are extracted and a previously-built recommender is executed. As mentioned in Section 3.2, the result offered by the recommender is either the best expected algorithm or a ranking of algorithms, depending on their suitability for creating a mining model that better fits the input dataset. It must be highlighted that "fits" must be understood according to the criteria with which the recommender was built (i.e., best accuracy, best f-measure, etc.). If different recommenders with different approaches have been created by the experts for the same domain, the novice user can choose one of them to be applied.

This process is implemented within a scientific workflow that can be used by novice data miners to load their datasets and to obtain the recommendation.

3.3.1. Scientific workflow for novice data miners

Our scientific workflow for novice data miners is shown in Fig. 9. It was designed for the novice data miners to learn how mining algorithms behave when applied on their datasets. This workflow has been designed by using MyExperiment.[14] Interestingly, the whole process is transparent to the user, since the workflow is the responsible for invoking the mining recommender, as well as using the datasets required by the user, thus returning the ranking of algorithms according to the design criteria of the recommender (e.g., sorted by predicted accuracy that would achieve the model by applying different mining algorithms[15]).

The workflow starts when a novice data miner loads input datasets into the system (*Load_file* and *send_arff _server*). Then, the meta-features of the input dataset is measured (*REST _MetaFeatures*). In the *Select _DataMining$_T$ask* box, a training model is obtained from the data stored in the knowledge base. Also, novice data miners may indicate which are their mining requirements (i.e., which is the goal to be achieved when data is analyzed). These mining requirements must be easily obtained from the novice data miner by means of a set of questions (boxes *Question_Level*1, *Question_Level*2, *Question_Level*3 and *Question_Level*4). Questions are based on a taxonomy (developed in our previous work [59]) that allows the scientific process to guide the novice data miner in deciding the data mining technique to use. Table 1 shows sample questions for data mining classification techniques.

With this taxonomy if, for example, the novice user wants to obtain a data mining model which makes a value prediction over an attribute (question 2), and this attribute is nominal (question 3), then it should be applied a classification data mining algorithm. After determining mining requirements, a recommender is executed by using the information from the models in the knowledge base, as well as the meta-features of the input dataset. The output of this scientific workflow is the algorithm recommendation (*AlgorithmRecommendation* box).

4. Experimental evaluation

In this section we explain how our framework was configured for building two data-mining classification algorithms' recommenders. The first one aims to predict students' performance in e-learning courses by using a collection of datasets from Moodle, thus a domain-based recommender; while the second one is a general-purpose recommender built from a set of datasets from the UCI repository.[16] Furthermore, with the aim of showing its applicability and generality, the first data-mining algorithm recommender is built by using regression techniques and its recommendation is based on the accuracy of the classifiers, whereas the second one, is built by using classification algorithms and its recommendation is based on the f-measure metric. Likewise, different meta-features are used to characterize original datasets according to the data types that they present. To describe each experiment, first an overview of datasets and meta-features used to feed our knowledge base is described; next, the setting chosen for each scientific workflow is explained and finally, the performance of each recommender is shown. The performance of these recommenders is carried out by comparing its answer with the one given by an expert data miner.[17]

[13] http://www.eclipse.org/acceleo.
[14] http://www.myexperiment.org/workflows/3846.html.
[15] http://www.myexperiment.org/users/16696.html.
[16] https://archive.ics.uci.edu/ml/datasets.html.
[17] Authors of the paper play the role of expert data miners

```
@relation KnowledgeBaseRandomForest
\%Complexity measures:
@attribute F1 numeric
@attribute F1v numeric
@attribute F2 numeric
@attribute F3 numeric
[..]
\%Simple measures:
@attribute numIns numeric
@attribute numClass numeric
@attribute numAtt numeric
@attribute numClass1 numeric
[...]
\%Statistical measures:
@attribute skAvg numeric
@attribute skMax numeric
@attribute skMin numeric
@attribute kurtAvg numeric
[...]
\%Landmarkers:
@attribute KNN1 numeric
@attribute BestNode numeric
@attribute RandomNode numeric
[..]
\%Accuracy of RandomForest:
@attribute RandomForest numeric
@data
1.826,5.845,0,0.331,0.601,0.404,0.141,0.5,0.173,0.423,0.117,[...]
0.037,0.042,0.000004,0.056,0.099,0.788,0.394,0.5,0.634,1.23,[...]
[...]
```

Code. 1. Excerpt of a sample derived *.arff* code.

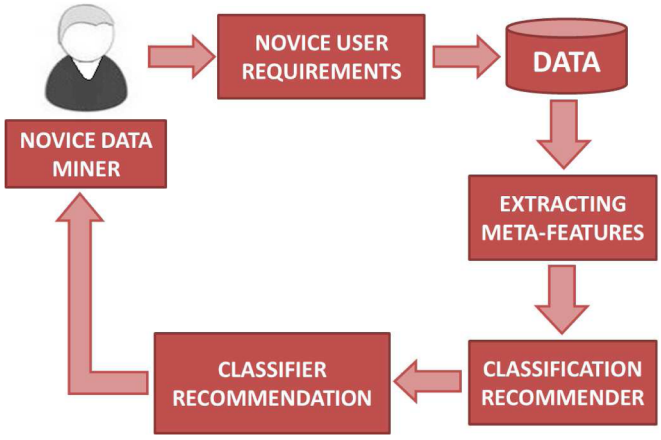

Fig. 8. Using the knowledge base for the data mining recommender.

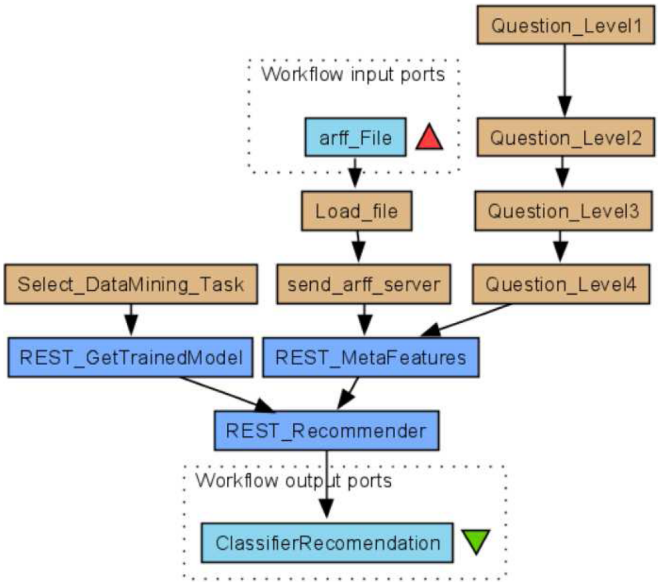

Fig. 9. Scientific workflow for novice data miners.

4.1. Experimentation with datasets from the educational domain

This experiment uses 30 datasets that gather the activity performed by students in virtual and blended courses hosted in a Moodle e-learning platform. This activity is measured by means of several metrics such as the total number of sessions open by each student in the course and in each tool of the course (tests, contents, forum, etc.), the number of self-tests performed, the number of messages posted and answered in the forum, among others. All attributes are numeric, except the class attribute which collects if the learner failed (positive class) or passed (negative class) the course. The size of these datasets ranges from 13 to 504 instances and from 3 to 28 attributes. Next, we describe how the workflows of our framework were setting to build this first algorithms' recommender, based on meta-regressors.

4.1.1. Creating and feeding the knowledge base

As mentioned in Section 3.1, the creation and feeding of the knowledge base (KB) is a task that must be performed by an expert in data mining. Expert data miner will be in charge of selecting and preprocessing the datasets before uploading to the KB, selecting the classifiers, the meta-features to be applied and the performance measures from which, later, the algorithms' recommender for novice users will be built.

In this experiment, we select eleven classifiers (offered by the Weka data mining tool [60]) to be applied to input datasets, leaving their default settings: C4.5, Ripper, OneRule, RandomForest, Cart, Ridor,

Table 1
Sample questions to guide novice data miner to select a mining technique.

Level	Question	Answer	Sample result
1	"What action do you want to perform on your data?"	Predict the value of an attribute that matches with the real answer	Predictive model
2	"Your target variable is numerical or nominal?"	Nominal type	Attribute to predict (e.g., a mark obtained by one student in a course), and these values are represented by strings (e.g., "PASS" or "FAIL")
3	"Show list of nominal attributes from the data source"	One of the attributes of nominal type of the data source.	Actual final mark

LogisticRegression, BayesianNetwork, NearestNeighbours with Generalization (NNge), Adaboost with DecisionStump and Bagging with DecisionStump. These were mainly selected for two reasons: (i) belong to different learning paradigms, and (ii) generate models easy to interpret for a novice data miner. It must be pointed out that the framework only includes white-box algorithms to meet this last requirement.

Next, from all the quality metrics available, we chose accuracy which returns the fraction of predictions that our model got right. It is a measure easy to understand and it is usually the first one in being evaluated in mining problems. Furthermore, it is the most frequently used metric in the learning domain as pointed out by Pea-Ayala [5].

Following, the method to measure the performance of each mining model (experiment) must be selected [61]. Different approaches to validate and evaluate the mining models can be applied. One of them is the hold-out process, which consists on dividing the dataset in two sets, the training set and the test set. However, the main problem with the hold-out process is that its evaluation results can be biased. To solve this problem, different approaches can be used, such as repeated-hold-out or the well-known k-cross-validation method. The k-cross-validation technique is used to reduce the bias of the evaluation measures by applying a training and testing process iteratively k times without overlapping, meaning that each instance is only used in the test set in a single iteration, and in the training set in the rest $k-1$ iterations. Finally, the evaluation measures obtained in each iteration, for example the accuracy, are averaged.

In this case, as a consequence of the reduced number of instances that the datasets include, a special variant of cross-validation, the leave-one-out cross-validation strategy, is chosen. With this strategy, the number of iterations is equal to the number of instances of the dataset, meaning that in each iteration, only one instance is used as test, and all the rest are used as training. Although it is well-known that this strategy is computationally demanding, it is the more suitable in our experiment for the sake of achieving generalization [61].

Finally, the datasets meta-features must be extracted and stored. As a consequence of the fact that all attributes in these datasets are numeric, the following ones were computed: the number of attributes, the number of instances and its dimensionality; the fourteen complexity meta-features offered by DCoL software whose description is included in Appendix B and the accuracy of five landmakers. These represent the performance achieved by weak classifiers which are useful to predict the performance of strong classifiers. For this case were used Linear-Discriminant (LD), BestNode with gain ratio criterion (BN), RandomNode (RN), NaiveBayes (NB) and 1-NN (Nearest Neighbours with k = 1). As it can be observed, these landmarkers were built with other classifiers than those used to build the meta-regressor.

4.1.2. Building and evaluating the recommender

Once our KB was fed with the 330 generated mining models (11 classifiers × 30 datasets) and the meta-features computed from these datasets, we ran the recommender system construction workflow. As it has been commented in Section 3.2, at this point the expert user must choose the supervised algorithm to build the recommender. In this experiment, a regressor will be chosen. The working-process of a regression-based recommender system is shown in Fig. 10. For each algorithm that the expert miner includes in the recommender, the workflow builds a meta-regressor which predicts the expected performance of this

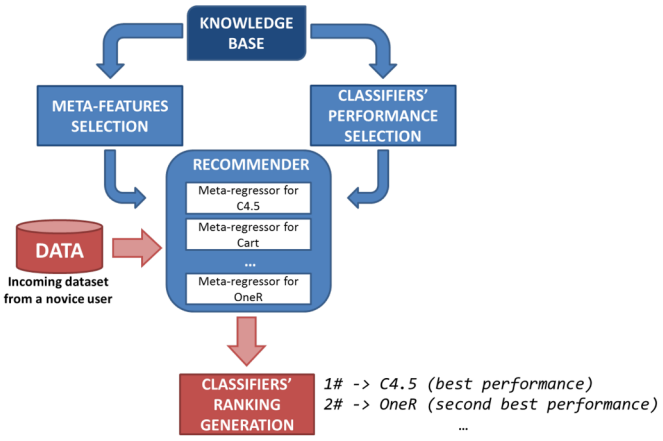

Fig. 10. Working process of a recommender built by means of meta-regressors.

algorithm for a new incoming dataset provided by a novice user from the meta-features extracted from it. Then, once the system computes the answer of all meta-regressors, it generates a ranking of the classifiers for the dataset at hand sorted by its performance, and finally chooses the one with the highest expected performance.

The meta-regressors can be built by means of different regression algorithms (such as Linear Regression, Support Vector Machines or Neural Networks, among others) using one of the performance measures available in the KB for these algorithms and datasets (for instance, accuracy, specificity, sensitivity or f-measure). The final decision will be made by the expert miner.

In this experiment, we used two different regression algorithms, Linear Regression and Multi-Layer Perceptron. Finally, we show here the results achieved with Linear Regression, which formula is shown in Appendix C, because these were quite similar to the ones got with neural networks and this technique is computationally less demanding and more interpretable. The performance measure used was accuracy.

In order to evaluate the efficiency of the recommendations made by S3Mining, we utilized the same 30 datasets applying a leave-one-out process. This means that, for each test dataset, we built a recommender system with the remaining 29 datasets. Thus, this process was repeated 30 times with the aim of assessing the recommender performance to select a good classifier for a novice user.

Fig. 11 shows the average Root-Mean Squared Error (RMSE) of the 11 meta-regressors used to predict the accuracy of each of the 11 classifiers calculated by following the aforementioned leave-one-out process. It can be observed that the RMSE is quite low in all cases ranging from 0.091 to 0.037. Therefore, it can be said that these meta-regressors have a high predictive power to choose the best classifier for future datasets.

If we sort these 11 classifiers by the accuracy they achieved for each dataset, the recommender selected one ranked in the first quartile for 17 out of 30 datasets (see Fig. 12). This means that the recommender chose a classifier with one of the highest accuracies in the 56.67% of times. Moreover, the best classifier was selected in 7 out of 30 datasets. We can also observe that for 25 datasets, more than the 83%, the classifier recommended one ranked in the first or second quartile and only selected

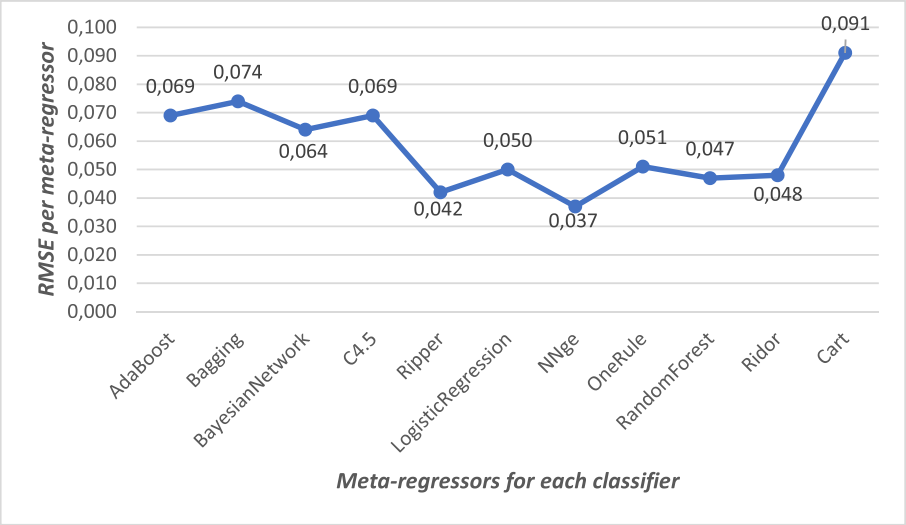

Fig. 11. RMSE per meta-regressor in the recommender system.

one with lower accuracy in 2 cases, representing the 6.67%.

Next, we show the recommendation and the ranking reached by the classifiers for 2 of the 30 datasets tested, which are named dataset1 and dataset2 in Tables 2 and 3 respectively. Columns "Pred. Acc." and "Pred. Rank" mean the real accuracy achieved by the classifiers and their real position in the ranking, whereas columns "Pred. Acc." and "Pred. Rank" represent the predicted accuracy of the classifiers and their position according to the output given by the recommender system.

As it can be observed in Table 2, the recommender chose LogisticRegression algorithm as classifier for the dataset1, which is, in fact, the classifier with the second best real accuracy. Moreover, it is worth point ing out that the classifiers which achieved the worst real accuracy for this dataset1, OneRule, Adaboost, Ripper, C4.5 and Bagging, are ranked in the last positions. Table 3 gathers the same information for the dataset2. Again, the recommender selected a classifier, RandomForest, whose accuracy is the second highest, meanwhile the classifiers with lowest real accuracy are ranked at the bottom.

Table 2
Ranking of classifiers generated by S3Mininng for dataset1.

Classifier	Pred. Acc.	Real Acc.	Pred. Rank.	Real Rank.
LogisticRegression	0.8152	0.7969	1	2
BayesianNetwork	0.8125	0.7813	2	3
NNge	0.7992	0.8906	3	1
RandomForest	0.7961	0.7813	4	3
Cart	0.7678	0.7813	5	3
Ridor	0.7625	0.7656	6	6
Bagging	0.7623	0.7344	7	8
C4.5	0.7566	0.7500	8	7
Ripper	0.7460	0.7344	9	8
Adaboost	0.7394	0.7344	10	8
OneRule	0.7198	0.7344	11	8

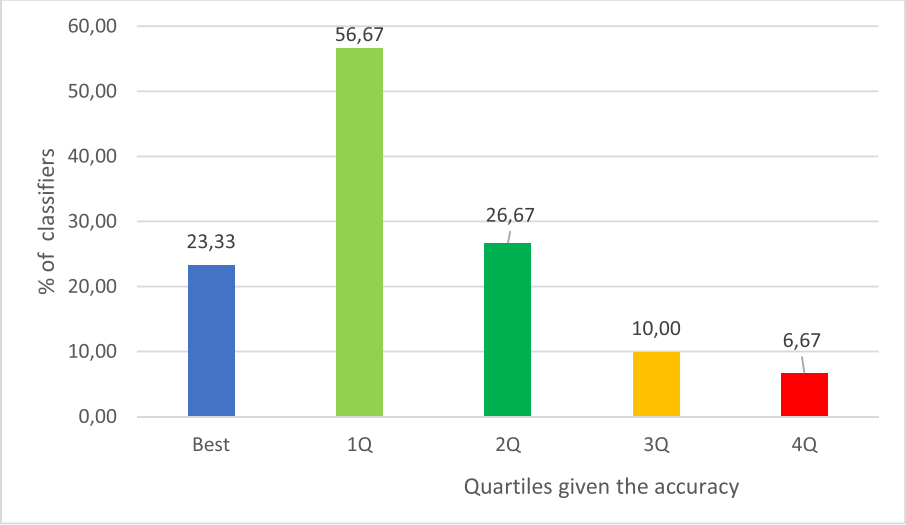

Fig. 12. Times that the meta-regressor chose a classifier whose accuracy corresponded to one belonging to that quartile.

Table 3
Ranking of classifiers generated by S3Mininng for dataset2.

Classifier	Pred. Acc.	Real Acc.	Pred. Rank.	Real Rank.
RandomForest	0.8824	0.8808	1	2
Bagging	0.8705	0.8808	2	2
Cart	0.8693	0.8757	3	6
AdaBoost	0.8677	0.8653	4	7
C4.5	0.8660	0.8808	5	2
Ripper	0.8657	0.8860	6	1
NNge	0.8656	0.8653	7	7
BayesianNetwork	0.8606	0.8806	8	5
OneRule	0.8603	0.8446	9	10
LogisticRegression	0.8246	0.8290	10	11
Ridor	0.8082	0.8549	11	9

4.2. Experimentation with datasets from UCI repository

This experiment uses 61 datasets from UCI repository,[18] which are briefly described in Appendix D. All of them were published to apply classification algorithms on them. These include nominal attributes and different number of classes to be predicted. The size of these datasets ranges from 10 to 10,992 instances and from 4 to 280 attributes. Next, we describe how the workflows of our framework wre setting to build this second algorithms' recommender based on meta-classifiers.

4.2.1. Creating and feeding the knowledge base

For this experiment, the workflow was run following the steps described in Section 3.1. The following 8 classifiers were selected leaving their default settings: Nearest Neighbours, LADTree, NaiveBayes-Tree, END, OneRule, C4.5, Nearest Neigbours with Generalization and AdaBoost with DecisionStump. These were chosen with the same criteria as in the previous experiment.

The quality metrics utilized in this experiment was f-measure, which is the harmonic mean of the precision and the recall or sensitivity. We chose it because, as accuracy, it is a measure easy to understand. Furthermore, this allows us to show the flexibility of our framework to build different recommenders. Again, a leave-one-out cross-validation strategy was used to calculate the f-measure of each classifier over each dataset.

In the last step, we selected the following meta-features: degree of class-unbalance with chi-square, entropy of numerical attributes and entropy of nominal attributes as statistical measures suited for nominal attributes; number of attributes, classes and instances as simple measures; and as landmarkers, the accuracy obtained by DecisionStump (DS), NaiveBayes (NB) and 1-NN (Nearest Neighbours with $k = 1$) algorithms.

4.2.2. Building and evaluating the recommender

Once our KB was fed with the 488 generated mining models (8 classifiers $\times$ 61 datasets) and the meta-features computed from these datasets, we ran the recommender system construction workflow. At this point the expert user must choose the supervised algorithm to build the recommender as well as the performance measure to be predicted. In this study, we show the results achieved by building the meta-classifier with C4.5, a tree-based mining model that is easily interpretable and f-measure as performance measure to be predicted.

The working-process of a classification-based recommender system is shown in Fig. 13. It is quite different from the regression-based recommender built in the previous experiment, in where a meta-regressor must be constructed for each one of the classifiers. In this case, only a meta-classifier is needed. This meta-classifier has the meta-features of each dataset in the experiment as predictor attributes and as class value to predict the best classifier for each one of them, that means, the classifier with the highest performance. So, when a novice data miner provides the system with a new dataset, this recommender will directly output the algorithm that is expected to have the highest performance, instead of generating a ranking of classifiers.

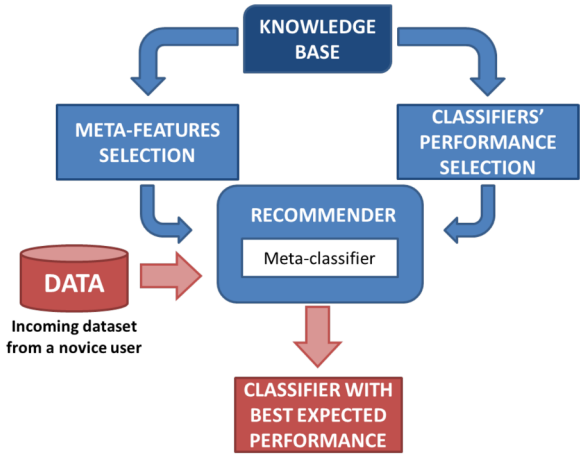

Fig. 13. Working process of the recommender built by means of meta-classifiers.

In order to evaluate the efficiency of the recommendations made by S3Mining, we utilized the same 61 datasets applying a leave-one-out process. This means that, for each test dataset, we built a recommender system with the remaining 60 datasets. Thus, this process was repeated 61 times with the aim of assessing the recommender performance to select a good classifier for a novice data miner.

Fig. 14 displays the f-measure achieved by the best classifier per dataset versus the f-measure obtained by the classifier recommended by our system. As it can be observed, the recommended classifier has a f-measure quite close to the best for the majority of datasets, except for 7 cases which represents only 11%.

The effectiveness of our approach can be better seen in Fig. 15. If we sort the 8 classifiers by the f-measure that they obtained for each dataset, the recommender selected one ranked in the first or second quartile for 36 out of 61 datasets (59.02% of times). Moreover, the best classifier was selected in 11 out of 61 datasets, and one of the first quartile in 20 out of 61. Only in 7 cases a dataset of the fourth quartile was chosen.

5. Conclusions and future work

The application of data mining techniques is commonly known as a hard process generally based on trial and error empirical methods. As a consequence of this fact, they can only be applied by data mining experts. In this paper, model-driven engineering and scientific workflow standards and tools are used for defining the S3Mining framework in order to support novice data miners in the application of classification algorithms over their structured data. Specifically, the following contributions have been achieved in this paper:

1. A metamodel that contains those useful concepts for representing models representing experiment data mining meta-features as a knowledge base.
2. Scientific workflows (that includes model-driven transformations) for (i) providing a mechanism for expert data miners to easily obtain all the information to automatically create and feed the knowledge base, as well as (ii) for offering novice data miners a mechanism to obtain a recommendation of the data mining algorithms to use on their datasets.
3. A set of experiments addressed to build recommenders are shown as proof of feasibility of our framework.
4. A public implementation of the workflows in Taverna, which are available on the Web.[19]

[18] https://archive.ics.uci.edu/ml/datasets.html.

[19] http://www.myexperiment.org/users/workflows/16696.

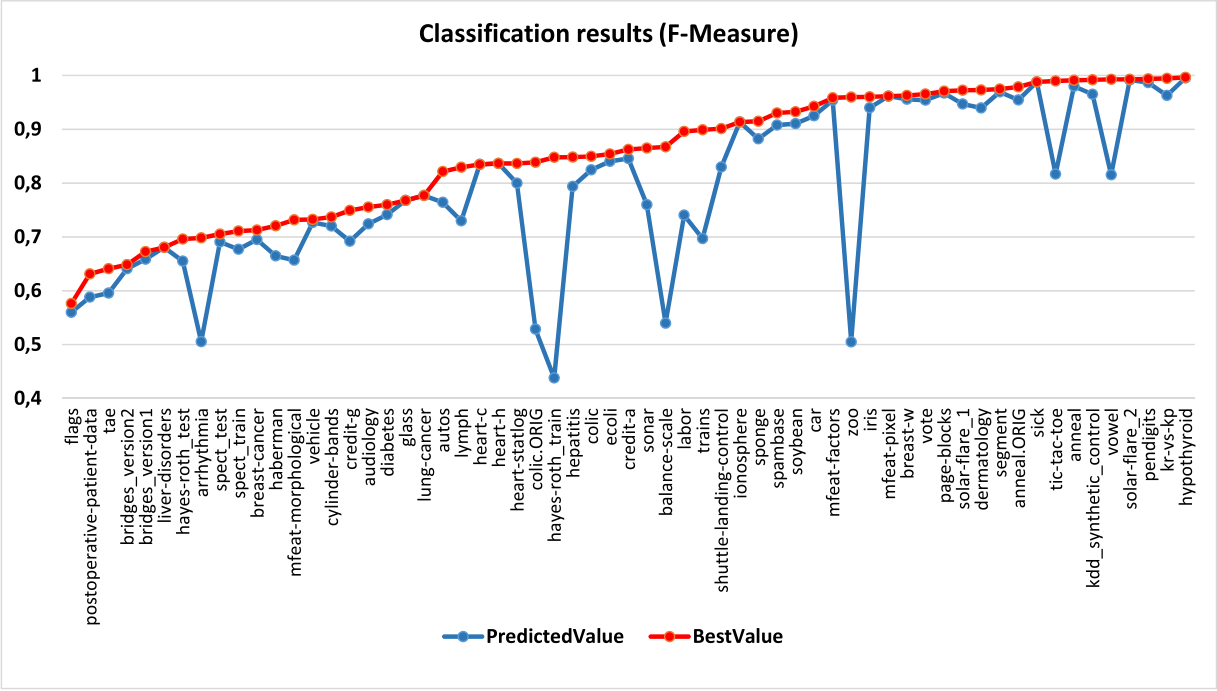

Fig. 14. Real f-measure vs predicted f-measure for each one of the 61 datasets.

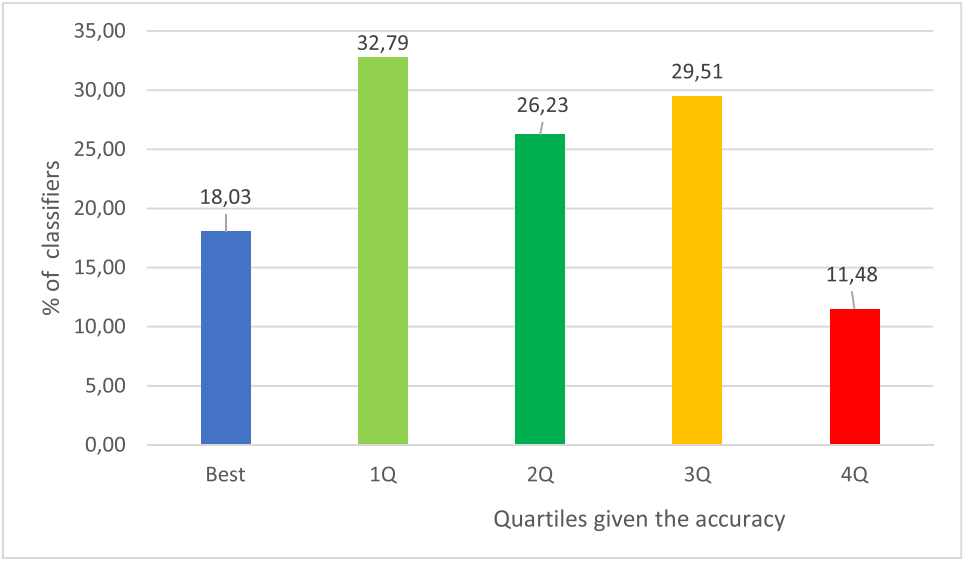

Fig. 15. Times that the meta-classifier chose a classifier whose accuracy corresponded to one belonging to that quartile.

Our knowledge base can be also useful as a resource for non-expert data miners to learn about algorithms' behavior and how the parameter tuning impacts on models.

As avenue for future work, we plan to study how recommenders can be included in our framework for other data mining algorithms apart from classification ones, e.g., for clustering [62] techniques. It is worth noting that this paper avoids dealing in detail with preprocessing techniques as we assume that the data sets are already preprocessed. As future work, we therefore consider different approaches of data-preprocessing and how they can be included in our framework. Other interesting challenge is the study of different types of recommendation approaches, like collaborative filtering [63] or trust-based recommendation [64]. Finally, the development of crowdsourcing techniques in order to collaboratively create the knowledge base is also planned as future work (e.g., in the sense of [65] that proposes the application of crowdsourcing and some techniques from open innovation to the scientific method for creating a collective intelligence), since it includes many challenges, as stated by Auer and Mann [17], such as encouraging researchers to create training data needed to automate the generation of knowledge bases in different fields of research.

Declaration of interests

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.

Acknowledgements

This work has been partially funded by Spanish Government through the research projects TIN2017-86520-C3-3-R and TIN2016-78103-C2-2-R.

Appendix A. DMKB metamodel class specification

A detailed description of each of the classes that compose the DMKB metamodel is presented in this appendix.

DMKB. This is the main class that represents the Data Mining Knowledge Base. From the relationship hasModels contains elements of type `DMKBModel`.
DMKBModel. This is the class that contains the useful elements for representing a Data Mining Knowledge Base (DMKB). The `DMKBModel` is a class that gathers all the information that is generated after analyzing a new data source. It collects the specification of a model in which the following information can be stored: input datasets, metafeatures, data mining algorithms, parameter-setting and data mining performance metrics.
DataSet. It describes the dataset used for generating the information included in the knowledge base. Each `DataSet` is composed of different fields. Each dataset belongs to a domain and contains a set of dataset metafeatures.
Field. It represents a piece of data contained in the `DataSet`. This piece of data is identified by a name. Also, the kind of field must be defined (by means of an enumeration called `FieldKind`) and its type (by means of an enumeration called `FieldType`). This class contains a set of metafeatures values that are related to the field.
FieldKind. It is an enumeration class for defining the type of value that the field instances may contain (continuous, categorical or mixed).
FieldType. It is an enumeration class for representing the type of each `Field` (numeric, date, nominal or string).
DataSetDomain. Domain which a specific dataset belongs to.
MiningModelPerformance. This class collects measures for evaluating the performance of each model (e.g. accuracy, f-measures, etc.).
Algorithm. This class represents information about the data mining algorithms that could be executed. Each algorithm belongs to a specific technique. (e.g. *NaiveBayes, J48, RandomTree* or *Adaboost*).
Parameter. It is a class that collects values of initial parameters when executing an algorithm. This class contains the name of parameter and its value.
Technique. This class defines a set of existing data mining approaches (e.g. tree-based classifiers, rule-based classifiers, distance-based clustering, and so on) inside each kind of problem. It contains a subgroup attribute in case that the algorithm requires to be further classified.
MiningTask. It defines the different kinds of data mining tasks (e.g. classification, prediction, clustering, etc.).
Metafeature. It is an abstract class that represents information related to the different criteria that can be presented either in a `DataSet` (`DatasetMetafeature`) or in each `Field` (`FieldMetafeature`). For each metafeature, a `ComputationMode` is defined to described how it is calculated (e.g. Pearson correlation method), and a `MeasureUnit` that represents the corresponding unit of measure if necessary.
DatasetMetafeature. This class inherits from the `Metafeature` class and collects values for each dataset metafeature defined.
FieldMetafeature. It inherits from the `Metafeature` class and gathers a value for specific `Field` class.
PredictedField. This class inherits from field and is designed to identify the target attribute that must be predicted when a type of problem is classification or regression.
ClassMeasuresValues. This class gathers the performance measures achieved by the mining model built for each value of the target attribute.

Appendix B. Complexity measures specification

Next, we provide a textual description of the complexity measures provided by DCol software[20] and used in our experiments as meta-features. A statistical description of these complexity measures can be found in [58].

F1: The maximum Fisher's discriminant ratio.
F1v: The directional-vector maximum Fisher's discriminant ratio.
F2: The overlap of the per-class bounding boxes.
F4: The maximum (individual) feature efficiency.
F4: The collective feature efficiency.
L1: The leave-one-out error rate of the one-nearest neighbor classifier.
L2: The minimized sum of the error distance of a linear classifier.
N1: The fraction of points on the class boundary.
N2: The ratio of average intra/inter class nearest neighbor distance.
N3: The training error of a linear classifier.
L3: The nonlinearity of a linear classifier.
T1: The fraction of maximum covering spheres.
T2: The average number of points per dimension.

Appendix C. Linear regression formula for building the meta-regressor models

The formula of the Linear Regression algorithm for building the meta-regressors of the experiment in Section 4.1 is shown in Eq. (C.1), where *PredictedEvalMeasure* is the evaluation measure to be predicted (e.g., Accuracy), M are the meta-features of the datasets used as predictor values, n is the number of meta-features, and w and b represent the regression coefficients.

$$PredictedEvalMeasure = w_1{*}M_1 + w_2{*}M_2 + \cdots + w_n{*}M_n + b \tag{C.1}$$

[20] https://github.com/nmacia/dcol.

Appendix D. Description of UCI datasets used in the experiments

Datasets from UCI repository[21] used in our experiments (described in Section 4.2) are shown in Table D.4. More information can be found in the UCI repository web.[22]

Table D.4
Description of UCI datasets used in the experiment in Section 4.2.

Dataset name	N# Att.	N# Inst.
heart-c	14	303
haberman	4	306
balance-scale	5	625
heart-h	14	294
spect_train	23	80
dermatology	35	366
lymph	19	148
labor	17	57
hepatitis	20	155
diabetes	9	768
bridges_version1	13	107
heart-statlog	14	270
sponge	46	76
tic-tac-toe	10	958
vowel	14	990
audiology	70	226
anneal	39	898
sonar	61	208
cylinder-bands	40	540
pendigits	17	10992
mfeat-pixel	241	2000
mfeat-factors	217	2000
zoo	18	101
bridges_version2	13	107
hayes-roth_train	5	132
vote	17	435
colic.ORIG	28	368
postoperative-patient-data	9	90
credit-a	16	690
liver-disorders	7	345
iris	5	150
autos	26	205
trains	33	10
flags	30	194
colic	23	368
ionosphere	35	351
sick	30	3772
kr-vs-kp	37	3196
spambase	58	4601
credit-g	21	1000
anneal.ORIG	39	898
hypothyroid	30	3772
lung-cancer	57	32
car	7	1728
shuttle-landing-control	7	15
breast-w	10	699
solar-flare_2	13	1066
spect_test	23	187
solar-flare_1	13	323
ecoli	8	336
kdd_synthetic_control	62	600
tae	6	151
page-blocks	11	5473
breast-cancer	10	286
soybean	36	683
mfeat-morphological	7	2000
arrhythmia	280	452
glass	10	214
vehicle	19	846
segment	20	2310
hayes-roth_test	5	28

[21] https://archive.ics.uci.edu/ml.
[22] https://archive.ics.uci.edu/ml/datasets.html.

References

[1] S. Newell, M. Marabelli, Strategic opportunities (and challenges) of algorithmic decision-making: a call for action on the long-term societal effects of datification, J. Strateg. Inf. Sys. 24 (1) (2015) 3–14.

[2] F. Provost, T. Fawcett, Data science and its relationship to big data and data-driven decision making, Big Data 1 (1) (2013) 51–59.

[3] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, Knowledge discovery and data mining: towards a unifying framework, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA, (1996), pp. 82–88. http://www.aaai.org/Library/KDD/1996/kdd96-014.php .

[4] Y. Cao, J. Gao, D. Lian, Z. Rong, J. Shi, Q. Wang, Y. Wu, H. Yao, T. Zhou, Orderliness predicts academic performance: behavioural analysis on campus lifestyle, J. R. Soc. Interface 15 (146) (2018), https://doi.org/10.1098/rsif.2018.0210.

[5] A. Pea-Ayala, Educational data mining: a survey and a data mining-based analysis of recent works, Expert Syst. Appl. 41 (4, Part 1) (2014) 1432–1462, https://doi.org/10.1016/j.eswa.2013.08.042.

[6] C. Romero, S. Ventura, Educational data mining: a review of the state of the art, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 40 (6) (2010) 601–618, https://doi.org/10.1109/TSMCC.2010.2053532.

[7] J. Gao, T. Zhou, Big data reveal the status of economic development, J. Univ. Electron. Sci. Technol. China 45 (4) (2016) 625–633.

[8] D. Huang, H. Zareipour, W.D. Rosehart, N. Amjady, Data mining for electricity price classification and the application to demand-side management, IEEE Trans. Smart Grid 3 (2) (2012) 808–817, https://doi.org/10.1109/TSG.2011.2177870.

[9] P. Lima-Monteiro, M. Zanin, E.M. Ruiz, J.P. Pimentão, P.A. da Costa Sousa, Indoor temperature prediction in an IOT scenario, Sensors 18 (11) (2018) 3610, https://doi.org/10.3390/s18113610.

[10] R. Nisbet, J. Elder, G. Miner, Handbook of Statistical Analysis and Data Mining Applications, Academic Press, 2009.

[11] J. Vanschoren, H. Blockeel, Stand on the shoulders of giants: towards a portal for collaborative experimentation in data mining, International Workshop on Third Generation Data Mining at the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 1 (2009), pp. 88–89.

[12] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Trans. Evol. Comput. 1 (1) (1997) 67–82.

[13] H.-P. Kriegel, K.M. Borgwardt, P. Kröger, A. Pryakhin, M. Schubert, A. Zimek, Future trends in data mining, Data Min. Knowl. Discov. 15 (1) (2007) 87–97.

[14] J. Kietz, F. Serban, S. Fischer, A. Bernstein, Semantics inside!, but let's not tell the data miners: Intelligent support for data mining, 11th International Conference on The Semantic Web: Trends and Challenge, (2014), pp. 706–720.

[15] A. Kalousis, M. Hilario, Model selection via meta-learning: a comparative study, 12th IEEE International Conference on Tools with Artificial Intelligence, (2000), pp. 406–413.

[16] R. Vicente-Saez, C. Martinez-Fuentes, Open science now: a systematic literature review for an integrated definition, J. Bus. Res. 88 (2018) 428–436.

[17] S. Auer, S. Mann, Toward an open knowledge research graph, Ser. Libr. (2018) 1–7 https://www.tandfonline.com/doi/full/10.1080/0361526X.2019.1540272.

[18] Y. Kim, J.M. Stanton, Institutional and individual factors affecting scientists' data-sharing behaviors: a multilevel analysis, J. Assoc. Inf. Sci. Technol. 67 (4) (2016) 776–799.

[19] B. Bilalli, A. Abelló, T. Aluja-Banet, R. Wrembel, Intelligent assistance for data preprocessing, Comput. Stand. Interfaces 57 (2018) 101–109.

[20] K. Kirchner, J. Zec, B. Delibašić, Facilitating data preprocessing by a generic framework: a proposal for clustering, Artif. Intell. Rev. 45 (3) (2016) 271–297, https://doi.org/10.1007/s10462-015-9446-6.

[21] M. Rönkkö, J. Heikkinen, V. Kotovirta, V. Chandrasekar, Automated preprocessing of environmental data, Future Gener. Comput. Syst. 45 (2015) 13–24.

[22] J. Whittle, J. Hutchinson, M. Rouncefield, The state of practice in model-driven engineering, IEEE Softw. 31 (3) (2014) 79–85.

[23] M. Brambilla, J. Cabot, M. Wimmer, Model-driven software engineering in practice, Synth. Lect. Softw. Eng. 1 (1) (2012) 1–182.

[24] A. Barker, J. Van Hemert, Scientific workflow: a survey and research directions, International Conference on Parallel Processing and Applied Mathematics, Springer, 2007, pp. 746–753.

[25] E. Deelman, D. Gannon, M. Shields, I. Taylor, Workflows and e-science: an overview of workflow system features and capabilities, Future Gener. Comput. Syst. 25 (5) (2009) 528–540.

[26] R. Espinosa, D. Garcia-Saiz, M.E. Zorrilla, J.J. Zubcoff, J. Mazón, Enabling non-expert users to apply data mining for bridging the big data divide, 3rd International Symposium on Data-Driven Process Discovery and Analysis - Revised Selected Papers, (2013), pp. 65–86.

[27] A. Dutt, M.A. Ismail, T. Herawan, A systematic review on educational data mining, IEEE Access 5 (2017) 15991–16005, https://doi.org/10.1109/ACCESS.2017.2654247.

[28] Y. Zhao, On interactive data mining, in: J. Wang (Ed.), Encyclopedia of Data Warehousing and Mining, IGI Global, 2009, pp. 1085–1090.

[29] T. Li, C. Zeng, W. Zhou, W. Xue, Y. Huang, Z. Liu, Q. Zhou, B. Xia, Q. Wang, W. Wang, X. Zhu, Fiu-miner (a fast, integrated, and user-friendly system for data mining) and its applications, Knowl. Inf. Syst. 52 (2) (2017) 411–443, https://doi.org/10.1007/s10115-016-1014-0.

[30] H. Dimitropoulos, H. Kllapi, O. Metaxas, N. Oikonomidis, E. Sitaridi, M.M. Tsangaris, Y. Ioannidis, Aition: a scalable platform for interactive data mining, Scientific and Statistical Database Management, Springer, 2012, pp. 646–651.

[31] S. Camiolo, A. Porceddu, Gff2sequence, a new user friendly tool for the generation of genomic sequences, BioData Min. 6 (1) (2013).

[32] S. De Bodt, D. Carvajal, J. Hollunder, J. Van den Cruyce, S. Movahedi, D. Inzé, Cornet: a user-friendly tool for data mining and integration, Plant Physiol. 152 (3) (2010) 1167–1179.

[33] M.E. Zorrilla, D. García-Saiz, A service oriented architecture to provide data mining services for non-expert data miners, Decis. Support Syst. 55 (1) (2013) 399–411.

[34] D. Garcia-Saiz, C. Palazuelos, M. Zorrilla, Data mining and social network analysis in the educational field: an application for non-expert users, in: A. Pea-Ayala (Ed.), Educational Data Mining, Studies in Computational Intelligence, 524 Springer International Publishing, 2014, pp. 411–439.

[35] A. Abelló, J. Darmont, L. Etcheverry, M. Golfarelli, J.-N. Mazón, F. Naumann, T.B. Pedersen, S. Rizzi, J. Trujillo, P. Vassiliadis, G. Vossen, Fusion cubes: towards self-service business intelligence, Int. J. Data Warehouse. Min. 9 (2) (2013) 66–88.

[36] P. Panov, L.N. Soldatova, S. Dzeroski, Towards an ontology of data mining investigations, Discovery Science, (2009), pp. 257–271.

[37] L. Soldatova, R. King, An ontology of scientific experiments, J. R. Soc. Interface 3 (11) (2006) 795–803.

[38] M. Hilario, A. Kalousis, P. Nguyen, A. Woznica, A data mining ontology for algorithm selection and meta-mining, ECML/PKDD09 Workshop on Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery, SoKD-09, (2009), pp. 76–87.

[39] M. Záková, P. Kremen, F. Zelezný, N. Lavrac, Automating knowledge discovery workflow composition through ontology-based planning, IEEE Trans. Autom. Sci. Eng. 8 (2) (2011) 253–264.

[40] J.-U. Kietz, F. Serban, A. Bernstein, S. Fischer, Designing KDD-workflows via HTN-planning, in: L.D. Raedt, C. Bessière, D. Dubois, P. Doherty, P. Frasconi, F. Heintz, P.J.F. Lucas (Eds.), ECAI, Frontiers in Artificial Intelligence and Applications, 242 IOS Press, 2012, pp. 1011–1012.

[41] C. Diamantini, D. Potena, E. Storti, Ontology-driven KDD process composition, International Symposium on Intelligent Data Analysis, (2009), pp. 285–296.

[42] M. Hilario, P. Nguyen, H. Do, A. Woznica, A. Kalousis, Ontology-based meta-mining of knowledge discovery workflows, in: N. Jankowski, W. Duch, K. Grabczewski (Eds.), Meta-Learning in Computational Intelligence, Studies in Computational Intelligence, 358 Springer, 2011, pp. 273–315.

[43] J. Vanschoren, L. Soldatova, Exposé: an ontology for data mining experiments, International Workshop on Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery, (2010), pp. 31–46.

[44] J. Vanschoren, H. Blockeel, B. Pfahringer, G. Holmes, Experiment databases—a new way to share, organize and learn from experiments, Mach. Learn. 87 (2) (2012) 127–158.

[45] H. Blockeel, J. Vanschoren, Experiment databases: towards an improved experimental methodology in machine learning, Knowledge Discovery in Databases: PKDD 2007, Lecture Notes in Computer Science 4702 Springer Berlin/Heidelberg, 2007, pp. 6–17.

[46] M. Hilario, e-LICO Annual Report 2010, Technical Report, Université de Geneve, 2010.

[47] D. Michie, D.J. Spiegelhalter, C.C. Taylor, J. Campbell (Eds.), Machine Learning, Neural and Statistical Classification, Ellis Horwood, Upper Saddle River, NJ, USA, 1994.

[48] B. Pfahringer, H. Bensusan, C. Giraud-carrier, Meta-learning by landmarking various learning algorithms, Proceedings of the 17th International Conference on Machine Learning, (2000), pp. 743–750.

[49] M. Hilario, A. Kalousis, Building algorithm profiles for prior model selection in knowledge discovery systems, Eng. Intell. Syst. 8 (2002) 956–961.

[50] M.E. Zorrilla, D. García-Saiz, Meta-learning: can it be suitable to automatise the KDD process for the educational domain? International Conference on Rough Sets and Intelligent Systems Paradigms, (2014), pp. 285–292.

[51] M. Charest, S. Delisle, Ontology-guided intelligent data mining assistance: combining declarative and procedural knowledge, Artificial Intelligence and Soft Computing, (2006), pp. 9–14.

[52] J.-U. Kietz, F. Serban, S. Fischer, A. Bernstein, "semantics inside!" but let's not tell the data miners: intelligent support for data mining, in: V. Presutti, C. d'Amato, F. Gandon, M. d'Aquin, S. Staab, A. Tordai (Eds.), The Semantic Web: Trends and Challenges, Springer International Publishing, Cham, 2014, pp. 706–720.

[53] C. Clifton, B. Thuraisingham, Emerging standards for data mining, Comput. Stand. Interfaces 23 (3) (2001) 187–193.

[54] R. Vilalta, Y. Drissi, A perspective view and survey of meta-learning, Artif. Intell. Rev. 18 (2002) 77–95.

[55] S. Segrera, J. Pinho, M.N. Moreno, Information-theoretic measures for meta-learning, Proceedings of the 3rd international workshop on Hybrid Artificial Intelligence Systems, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 458–465, https://doi.org/10.1007/978-3-540-87656-4_57.

[56] Y. Peng, P.A. Flach, C. Soares, P. Brazdil, Improved dataset characterisation for meta-learning, in: S. Lange, K. Satoh, C.H. Smith (Eds.), Discovery Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2002, pp. 141–152.

[57] G.D.C. Cavalcanti, T.I. Ren, B.A. Vale, Data complexity measures and nearest neighbor classifiers: a practical analysis for meta-learning, 2012 IEEE 24th International Conference on Tools with Artificial Intelligence, 1 (2012), pp. 1065–1069, https://doi.org/10.1109/ICTAI.2012.150.

[58] T.K. Ho, M. Basu, M.H.C. Law, Measures of Geometrical Complexity in Classification Problems, Springer London, London, pp. 1–23.

[59] R.E. Oliva, J.J. Zubcoff, J. Mazón, Requirements taxonomy for automatic detection

of data mining techniques for non-expert users, 35th International Conference of the Chilean Computer Science Society, (2016), pp. 1–9.
[60] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (2009) 10–18 http://doi.acm.org/10.1145/1656274.1656278.
[61] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2000.
[62] D.G. Ferrari, L.N. de Castro, Clustering algorithm selection by meta-learning systems: a new distance-based problem characterization and ranking combination methods, Inf. Sci. 301 (2015) 181–194.
[63] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 5–53.
[64] L.-J. Chen, J. Gao, A trust-based recommendation method using network diffusion processes, Physica A 506 (2018) 679–691.
[65] T. Bücheler, J.H. Sieg, Understanding science 2.0: crowdsourcing and open innovation in the scientific method, Procedia Comput. Sci. 7 (2011) 327–329.