Enabling Non-expert Users to Apply Data Mining for Bridging the Big Data Divide

Roberto Espinosa[1], Diego García-Saiz[2], Marta Zorrilla[2], Jose Jacobo Zubcoff[3], and Jose-Norberto Mazón[4(✉)]

[1] WaKe Research, Universidad de Matanzas "Camilo Cienfuegos", Matanzas, Cuba
respinosa@umcc.cu
[2] MatEsCo, Universidad de Cantabria, Santander, Spain
{diego.garcia,marta.zorrilla}@unican.es
[3] WaKe Research, Dept. Ciencias del Mar y Biología Aplicada, Universidad de Alicante, Alicante, Spain
jose.zubcoff@ua.es
[4] WaKe Research, Dept. Lenguajes y Sistemas Informáticos, Instituto Universitario de Investigación Informática, Universidad de Alicante, Alicante, Spain
jnmazon@dlsi.ua.es

Abstract. Non-expert users find complex to gain richer insights into the increasingly amount of available heterogeneous data, the so called big data. Advanced data analysis techniques, such as data mining, are difficult to apply due to the fact that (i) a great number of data mining algorithms can be applied to solve the same problem, and (ii) correctly applying data mining techniques always requires dealing with the inherent features of the data source. Therefore, we are attending a novel scenario in which non-experts are unable to take advantage of big data, while data mining experts do: the big data divide. In order to bridge this gap, we propose an approach to offer non-expert miners a tool that just by uploading their data sets, return them the more accurate mining pattern without dealing with algorithms or settings, thanks to the use of a data mining algorithm recommender. We also incorporate a previous task to help non-expert users to specify data mining requirements and a later task in which users are guided in interpreting data mining results. Furthermore, we experimentally test the feasibility of our approach, in particular, the method to build recommenders in an educational context, where instructors of e-learning courses are non-expert data miners who need to discover how their courses are used in order to make informed decisions to improve them.

Keywords: Knowledge base · Big data · Data mining · Recommender · Meta-learning · Model-driven development

1 Introduction

The increasing availability of data is a great opportunity for everyone to take advantage of their analysis. The "big data promise" states that the more data you

P. Ceravolo et al. (Eds.): SIMPDA 2013, LNBIP 203, pp. 65–86, 2015.
DOI: 10.1007/978-3-662-46436-6_4

have, the more analysis you can perform, and then, the more informed decisions you can make. Unfortunately, this could be only true for experts in data analysis (the so-called, data scientists) or for those companies that may hire them; but, what about non-experts data miners?[1] Physicians in hospitals, teachers in high schools or universities, and so on; would be interested in applying advanced data analysis techniques to make informed decisions in their daily life.

Importantly, data mining is one of the most prominent technique to discover implicit knowledge patterns, thus gaining richer insights into data. However, non-expert users may find complex to apply data mining techniques to obtain useful results, due to the fact that it is an intrinsically complex process [14,20] in which (i) a great number of algorithms can be applied to solve the same problem with different outcomes, and (ii) correctly applying data mining techniques always requires a lot of manual effort for preparing the datasets according to their features. Consequently, successfully applying data mining requires the know-how of an expert in order to obtain reliable and useful knowledge in the resulting patterns.

Democratization of data mining therefore requires relying on knowledge about suitable data mining techniques and settings according to their data features. User-friendly data mining [13] is a step forward to this democratization, since it fosters knowledge discovery without mastering concepts and data mining techniques, thus bridging the "big data divide" and allowing everyone to take advantage of the available big data.

In this paper we introduce our model-driven framework to allow non-expert users apply data mining in a user-friendly manner. It is based on a knowledge base on which a recommender will be built. Our framework makes use of different techniques and tools which are orchestrated by means of scientific workflows, in order to be easily replicated as well as enabling the extension of the knowledge base. In the previous version of this work [4], we presented a model-driven approach for creating and using this knowledge base. In this extended version, the contributions are: (i) a proposal for allowing non-expert users to specify data mining requirements without having extensive knowledge of data mining, (ii) a set of mechanisms for guiding non-experts users to interpret and used the data mining results, and (iii) a description of how the recommender is constructed. An overview of our approach is shown in Fig. 1.

We test our approach in an online educational context: instructors of e-learning courses are non-expert data miners who need to discover whom and how their courses are used in order to improve them. Data mining is being profusely used [17] in the educational context as consequence of the rapid expansion of the use of technologies in supporting learning. This is used in established institutional contexts and platforms, and also, in the emerging landscape of free, open, social learning online. Although there are tools as ElWM [26] which help instructors to analyse their virtual courses, a knowledge base as proposed here will become a crucial resource for designing a recommender that help instructors

[1] For us, a "non-expert user" is one who has basic knowledge of statistics but does not know how to apply data mining algorithms satisfactorily.

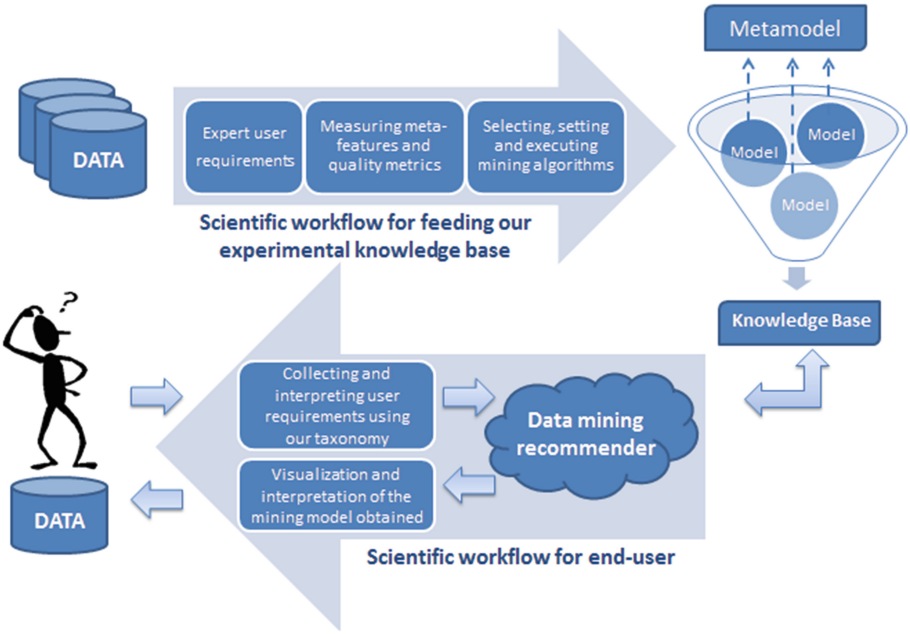

Fig. 1. Overview of the user-friendly data mining approach.

(as non-expert data miners) in applying the right data mining algorithm on their datasets and to extract conclusions oriented to improving the teaching-learning process.

The remainder of this work is structured as follows: an overview of the related work is presented in Sect. 2. Our approach is described in Sect. 3, while the conducted experiments are described in Sect. 4. Finally, conclusions and future work are sketched in Sect. 5.

2 Related Work

The data mining algorithm selection is at the core of the knowledge discovery process [5]. Several data mining ontologies have been developed to provide adequate knowledge to help in this selection. For example, OntoDM [15] is a top-level ontology for data mining concepts that describes basic entities aimed to cover the whole data-mining domain, while EXPO ontology [19] is focused on modeling scientific experiments. A more complete ontology is DMOP [9] 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 [25] and the eProPlan ontology [12].

Regarding data mining workflows, the KDDONTO ontology [3] aims at both discovering suitable KD algorithms and describing workflows of KD 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 [10] is aimed at supporting workflow construction for the knowledge discovery process. Moreover, in [22] 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 [21]).

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 [2] 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 [8] has 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 [10].

Unlike our proposal, both projects are oriented to support expert data miners. Our framework would help naive practitioners data miners and non-experts users to have a kind of guidance to obtain a mining result easily.

Furthermore, although ontologies used in the aforementioned approaches are very useful for providing semantics, they lack mechanisms for automating the management (and interchange) of metadata, such as metamodeling [16]. Metamodeling provides a common structure for storing the most relevant information in models, thus avoiding interoperability and compatibility problems. For example, having a metamodel allows us to specify data coming from different DBMS in a model which can be easily used as input data set for data mining experiments.

3 Knowledge-Based Approach for Enabling Non-expert Users to Apply Data Mining

Our approach aims to bridge the "big data divide" when advanced data analysis methods are used. In this section, we describe each of the steps included in our approach.

3.1 Allowing Non-experts to Specify Data Mining Requirements

Data mining is a complex process composed by a set of steps that must be applied to the data sources in order to discover knowledge. One of the reasons that hinders the application of data mining techniques is that non-experts users are unable to express their data mining requirements, i.e. what kind of knowledge they can discover from data.

With the aim of guiding non-expert users to specify their requirements and goals, we propose a taxonomy based on questions. Since non-expert users have no expertise on data mining techniques, our taxonomy fosters a friendly environment that allows them to transform their initial expectations in data mining requirements.

The elements that form the created taxonomy have been identified both from a theoretical detailed study and, from our own experience in the area. In this way, the taxonomy represents a structure that connects the identified concepts that are part of the knowledge discovery process with their possible values in each case. Also, this taxonomy aims to use a simple language, bearing in mind that its main users are not expert in data mining.

Requirements taxonomy is shown in Fig. 2. It has a tree structure, where questions that guide the data mining technique selection are represented as nodes and the possible answers are the respective arches that drive user to the following question. The leaf nodes represent the data mining technique that would be useful for the user.

Our taxonomy can be easily used by a non-expert user without knowing data mining concepts by means of the design of simple questions and answers.

The first step is selecting the data source that will be analyzed. Then the structure of the data source can be read, and the composition of the set of attributes is known. The format of the input data source could be an *.arff* file. Data mining techniques are grouped into two kind of models: predictive and descriptive. Predictive models intend to estimate future or unknown values of the interest variables. For example, a predictive model aims to estimate the category of the customers according to their frequent expenses at a supermarket. Descriptive models identify patterns that explain or summarize the data. For example, a supermarket desires to identify groups of people with similar preferences with the aim of organizing different offers for each group. If the user selects

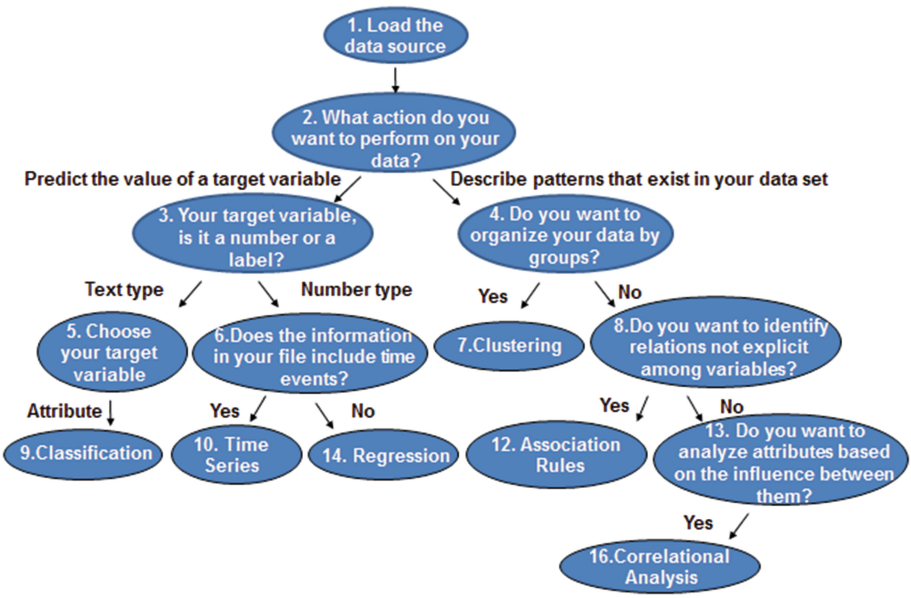

Fig. 2. Taxonomy for helping non-experts to specify data mining requirements.

a predictive model, the next question is focused on the target attribute data type that he wants to predict. If the information of the file that he wants to analyze include time events its highly probably that he wants to apply "Time Series". For example, to know an estimate of a company's sales in a next year, having a considerable amount of historical sales records. In the other case "Regression technique". In case the user selects a descriptive model, and he wants to organize data by groups, he must apply "Clustering technique". For example, if you want to know which are the most relevant features of your gold, silver and bronze customers according to their consume. If user is interested in identifying non explicit relationships among attributes, he must apply "Association Rules" techniques. The typical example, the market basket analysis, what items are frequently bought along with the beers?, is solved with these techniques. Finally, if user wants to analyze attributes, based on the influence between them, he must use "Correlational Analysis". Example: Is the learners' activity correlated with the mark, i.e. more activity implies more grade?

After using the taxonomy for determining the data mining technique to use, we have proposed a data mining knowledge base (and a method for creating and using it) that will be used to build a data mining algorithm recommender. Knowledge base and recommender are described in the next sections.

3.2 Data Mining Knowledge Base

Our knowledge base brings the results on executing data mining processes on many datasets. It can be therefore used as a resource to keep information about the behavior of different data mining algorithms with regard of the data sources quality and general characteristics of data set. To this aim, our knowledge base contains the following information:

General characteristics or features of input datasets. Metadata from the datasets must be known, as number of attributes and instances, as well as the corresponding data types.

Data quality features. Several quality criteria from the datasets must be measured. Quality criteria are directly related to datasets (e.g. percentages of null values), as well as fields (e.g. field correlation).

Results when applying a data mining algorithm. Some information related to the execution of a data mining algorithm is acquired: data mining technique being executed, predicted attribute and its results (quality of the model measured by accuracy, TPrate, F-score, and so on).

Scientific Workflows for the Development of Our Knowledge Base. The development of our data mining knowledge base is driven by the development of a scientific workflow. This workflow is in charge of (i) collecting all the required information for our knowledge base (as previously stated), and (ii) creating the knowledge base.

Fig. 3. Our Taverna workflow.

Scientific workflows are largely recognized as useful paradigms to describe, drive, and share information about experiments[2]. We used Taverna Workbench in our approach. This is a widely used open source Workflow Management System.

Our workflow (see Fig. 3) has as a main objective the data sets processing in order to create models to conform the knowledge base. To this end, the workflow begins with the loading of the data source (e.g., *.arff* files[3]) on which will be applied a set of data mining algorithms[4]. Next step is about to obtain a predicted attribute (usually the last column). All these results are part of the obtained model, and all data mining algorithms are executed, leading to a result set. Simultaneously, the workflow measures the quality criteria values of the data source according to some quality criteria. The workflow can be run manually or configured by command line. It is worth noting that our Taverna workflow is published at http://www.myexperiment.org/workflows/3843.html. Once the experiments on the data source are processed and evaluated, are stored in the knowledge base.

[2] http://en.wikipedia.org/wiki/Scientific_workflow_system.

[3] Attribute-Relation File Format (ARFF), a file format used by the data mining tool Weka [6] to store data.

[4] Our Taverna workflow was designed to be useful for any mining technique, but in this work we only consider classification techniques.

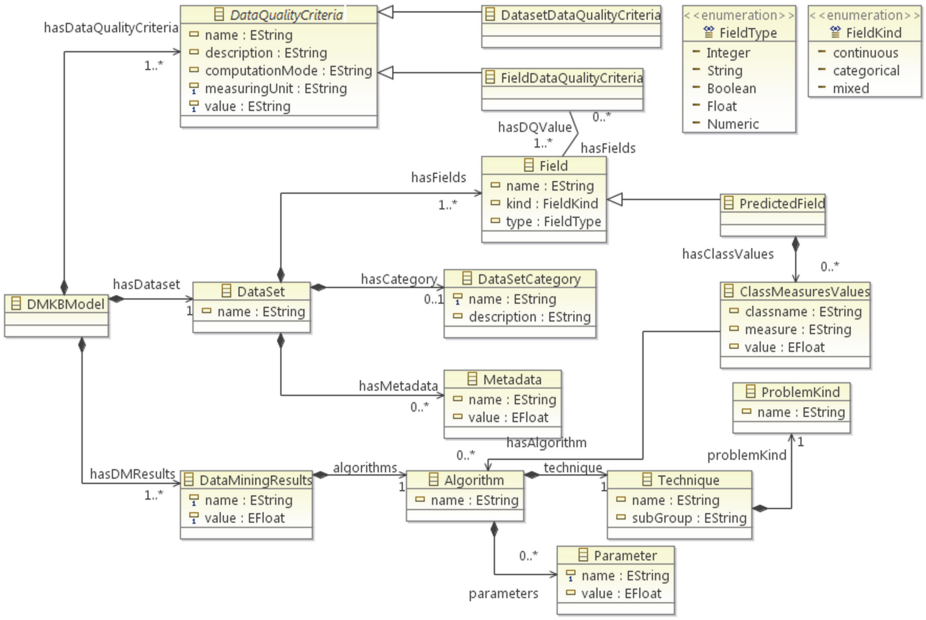

Fig. 4. Our metamodel for representing our data mining knowledge base.

Generating a Data Mining Knowledge Base. Our knowledge base aims to represent in a structured and homogeneous manner all the necessary data mining concepts. Once, the knowledge base is obtained the practitioner could use it to evaluate the real data set in our recommender (explained in Sect. 3.3) in order to obtain the adequate predicted model taking in account the data set features.

The aim of our metamodel is being as generic as possible. Therefore, any data related to the aforementioned information about data mining experiments (metadata of data sources, results of data mining algorithms, and values of data quality criteria) is adequately represented in a model. Our models are not restricted to a certain quality metrics or features, since the metamodel support creating new criteria in each model as required. The definition of our metamodel (see Fig. 4) is based on an analysis of several ontologies (see Sect. 2):

DMKBModel. This is the main class that contains the other useful elements for representing a Data Mining Knowledge Base (DMKB). The `DMKBModel` class allows the specification of a model in which the following information can be stored: input datasets, metadata, data mining algorithms, parameter-setting, data mining results generated when the Taverna workflow is executed, and data quality criteria.

DataSet. It describes datasets used for generating the information included in the knowledge base. Each `DataSet` is composed of different fields. Also, each data set contains a category and a set of metadata.

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 data quality values that are related to the field.

FieldKind. It is an enumeration class for defining the general kind of values that the field instances may have.

FieldType. It is an enumeration class for representing the type of each `Field`.

DataMiningResults. This class represents values of measures for each data set after executing an algorithm, e.g., accuracy.

Algorithm. This class represent information about executed data mining algorithms. Each algorithm belongs to a specific technique. E.g., *NaiveBayes*, *J48*, *RandomTree* or *Adaboost*.

Parameter. It is a class that represents values of initial parameters when executing an algorithm. This class contains the name of the parameter and a value.

Technique. This class defines a set of existing data mining techniques (e.g. a tree, a probability matrix, etc.). It contains a subgroup attribute in case that the algorithm requires to be further classified.

ProblemKind. It defines the different kinds of problem with which the user need is satisfied (e.g., classification, prediction, clustering, etc.).

DataQualityCriteria. It is an abstract class that represents information related to the different criteria that can be presented either in a `DataSet` (`DatasetDataQualityValue`) or in each `Field` (`FieldDataQualityValue`). For each data quality criteria, a `ComputationMode` is defined to described how it is calculated (e.g., Pearson correlation method), and a `MeasuringUnit` that represent the corresponding unit of measure.

DatasetDataQualityValue. This class inherits from the `DataQualityCriteria` class and defines data quality value criteria for a `Dataset`.

FieldDataQualityValue. It inherits from the `DataQualityCriteria` class and represents a value for specific `Field` class.

In order to store the mining results in the knowledge base conforming to the metamodel presented in Fig. 4 were developed the following transformations in Eclipse Framework[5]. These transformations are executed in Taverna by means of a web service.

Transformation tasks for generating models have been supported with the use of Java facilities provided by the Eclipse Modeling Framework (EMF)[6]. The Java code 1.1 shows an excerpt of the transformation in charge of creating a model within the knowledge base. For each of the data mining algorithms executed by the workflow, the following classes are generated: `DataMiningResult`, `Algorithm`, `Technique`, and `ProblemKind`; as well as the required existing relationships among them: `hasDMResults`, `algorithms`,

[5] http://www.eclipse.org.
[6] http://www.eclipse.org/emf.

Fig. 5. Sample model of comp2class data set.

`technique`, and `problemKind`. Finally, the model (represented by means of a XMI file) is created. Figure 5 shows a sample `DMKBModel` generated by using our approach.

```
for (int i = 0; i <= First.listaResAlg.size()-1;i++)
    {
        DataMiningResults dmr =kbf.createDataMiningResults();
        dmr.setName(First.listaResAlg.get(i).requirementName);
        dmr.setValue(First.listaResAlg.get(i).value);
        Algorithm alg= kbf.createAlgorithm();
        alg.setName(First.listaResAlg.get(i).algName);
        Technique tec=kbf.createTechnique();
        tec.setName(First.listaResAlg.get(i).technique);
        tec.setSubGroup(First.listaResAlg.get(i).subgroup);
        ProblemKind pk=kbf.createProblemKind();
        pk.setName(probKind);
        alg.setTechnique(tec);
        tec.setProblemKind(pk);
        dmr.setAlgorithms(alg);
        model.getHasDMResults().add(dmr);
    }
ResourceSet rs = new ResourceSetImpl();
rs.getResourceFactoryRegistry().getExtensionToFactoryMap().put ("xmi",
new XMIResourceFactoryImpl());
Resource resource=rs.createResource(URI.createFileURI("ouput_generated/"+
ds.getName()+".xmi"));
resource.getContents().add(model);
```

Code 1.1. Segment of Java code to create a model.

Our knowledge base is composed by the set of models obtained after running the Taverna workflow for each input data set. Once the knowledge base is obtained, it can be used for the construction of a recommender. It will be in charge of choosing the best mining algorithm for each data set taking into account its features.

3.3 Recommender System

In our proposal the knowledge base is used for the construction of the recommender, we delegate this task in an expert data miner because it is highly important and its precision strongly depends on the instances chosen, the algorithm used and its parameters setting. We rely on meta-learning to build our recommender since this technique has been demonstrated suitable to assist users to choose the best algorithm for a problem at hand [11,23].

To create this recommender, the expert must select the most suitable instances of the knowledge base. The expert must take into account the context of each data set and the target audience, for example teachers. In this case, only algorithms with an easily to interpret output can be used. Initially the expert will use all meta-features stored in the knowledge base but some of them could be eliminated during the process if they do not provide with significant information.

The meta-features used can be classified in three groups: general, quality-related and based on information theoretic features. In particular, we selected the number of attributes and instances in the data set, the number of categorical and numerical attributes, the type of data in the data set (numeric, nominal or mixed) and the number of classes. Regarding quality, we chose completeness (percentage of null values) and finally, we used class entropy in order to establish if the class was balanced or not. We defined three possible values for this attribute (balanced, quite unbalanced, highly unbalanced). Our recommender then can compare the values of these characteristics with those provided by a new data set, thus returning the most accurate algorithm to be executed against it. In our experimentation (see Sect. 4), we have built two ad-hoc recommenders for evaluating the feasibility of our approach.

3.4 Interpreting Data Mining Results for Non-experts

A specific scenario is used through this section: a teacher involved in virtual education. We focus on education as a consequence of the fact we have a rich knowledge base with instances of this domain [4]. Furthermore it is a field of great interest since, in one way or another affects a large part of society. Most educational institutions use an e-learning platform such Moodle or Blackboard to support distance education. These usually offer some tools to extract student activity data that teachers can use to generate the input file that they can use with our approach. We work with Moodle because it is one of the most used systems in this educational field.

Moodle provides a reporting tool which enables teachers to know some facts about the activity performed in the course. This activity can be filtered by learner, resource and dates. Thus, teachers can build the input file to our platform by performing several queries in this tool. Another alternative is to work directly on the database which collects this activity, which can be easily carried out by the system administrator.

Table 1 displays the attributes commonly used to analyze Moodle courses from an analytical point of view [18].

Table 1. Commonly used attributes extracted from Moodle to analyze student's performance.

Name	Description
Course	Identification number of the course
n_assigment	Number of assignments handed in
n_quiz	Number of quizzes taken
n_quiz_a	Number of quizzes passed
n_quiz_s	Number of quizzes failed
n_messages	Number of messages sent to the chat
n_messages_ap	Number of messages sent to the teacher
n_posts	Number of messages sent to the forum
n_read	Number or forum messages read
total_time_assignment	Total time spent on assignment
total_time_quiz	Total time used in quizzes
total_time_forum	Total time used in forum
final_mark	Mark the student obtained in the course

Our scenario comes from the activity performed by 432 students enrolled in a computer science course. In the code 1.2 a sample of the first instances in this file in *.arff* format is shown.

```
@relation final_nominal-weka.filters.unsupervised.attribute. Remove-R1-2
@attribute n_assignment numeric
@attribute n_assignment_a numeric
@attribute n_assignment_s numeric
@attribute n_messages numeric
@attribute n_posts numeric
@attribute n_read numeric
@attribute n_quiz numeric
@attribute n_quiz_a numeric
@attribute n_quiz_s numeric
@attribute n_messages_ap numeric
@attribute total_time_assignment numeric
@attribute total_time_quiz numeric
@attribute total_time_forum numeric
@attribute mark {PASS,FAIL}
@data
6,0,6,0,0,0,0,0,0,0,1872,0,404,FAIL
7,0,7,0,0,0,0,0,0,0,2163,0,-14429850,FAIL
6,0,6,0,0,0,0,0,0,0,1496,0,51,FAIL
7,0,7,0,0,0,0,0,0,0,1386,0,145,PASS
```

Code 1.2. An excerpt of the computer science course data file in *.arff* format.

As can be observed, the teacher's goal is to predict if a new student will pass or not the course by analyzing the activity performed by other students in a previous edition of the course (see target attribute, the last one in the aforementioned list). The use of taxonomy allows the system to know that user needs to apply a classification technique (i.e. predict by using an attribute).

Then, when the recommender system determines which is the best mining model to apply, it is executed and the pattern obtained as result is displayed. The outcome of our scenario can be observed in Fig. 6.

Teacher, observing this output, should assess the goodness of the model obtained. That means, user should check if the accuracy of the model is good enough to her/his goal. Accuracy is the most frequent metric to evaluate the quality of a prediction model. It must be understood as the percentage of success in the process of classifying the instances collected in the *.arff* file provided correctly. An accuracy near 50 %, in a two-class problem, is as flipping a coin and thus, the model should be discarded. In the opposite side, although it could be thought to be very good, a model with an accuracy higher than 95 % could be overfitted. Thus, an accuracy between 75 % and 90 % is commonly accepted as a good result when we work with real data. Of course, there are scientific techniques to compare different models but this is a task which requires the work of an expert data miner, so it is out of the scope of the goal of this paper, since our approach is focusing on avoiding the expert data miner implication. It must be reminded that the goal of our approach is to democratize the use of data mining, and of course, the outcomes which can be obtained by means of semi-automatics will never be able to rise the same accuracy that using manual procedures, but we skip the time-consuming and costly task of counting on an expert.

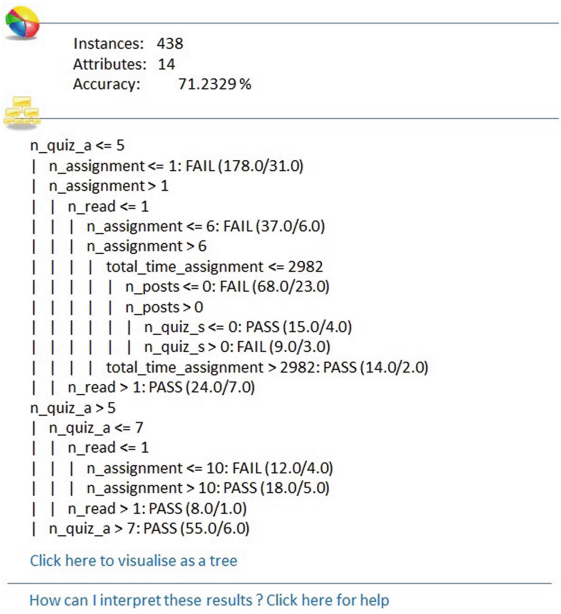

Fig. 6. Pattern obtained from Moodle course data.

The model generated in this study case is a J48 tree, one of the simplest and easiest to understand models. The end user only has to build a set of rules from the root of the tree to its leaves. For instance, some of rules extracted from our example are shown in code 1.3.

```
if ''Number of quizzes passed'' > 7
   then student passes
if ''Number of quizzes passed'' < 5
   and  ''Number of assignments performed'' > 1
   and ''student reads forums messages''
   then the student passes
if the student neither passes quizzes (n_quiz_a <= 5)
   nor carries out assignments (n_assignment <= 1)
   then student fails (as expected)
```

Code 1.3. Rules extracted from our sample.

The number of well-classified and bad-classified instances in that branch appear in the leaves. Thus, the user can assess the goodness of each rule and whether it should be considered or not following the criteria exposed previously.

Additionally, it must be highlighted that, sometimes the whole rule is not so interested to make decisions as knowing which attributes are the most significant contributors to classify students in this case. The most relevant are those which appears in the first levels of the tree, in this case, the number of quizzes passed and the number of assignments performed. This explanation is given to users when they click on the link *How can I interpret these results?*. This result could be also seen as a tree, as displayed in Fig. 7.

4 Experimental Evaluation

The knowledge base is the sustenance for building the recommender which selects the best classifier for an input test data set. Therefore, the goal of this experiment is twofold: on one hand, knowing if the generated knowledge base is suitable for the construction of the recommender, and on the other hand, evaluating the goodness of our recommender. The methodology followed comprises the steps listed below:

1. Selection of courses and data extraction from e-learning platforms.
2. Generation of 99 datasets as described in Sect. 4.1.
3. Using a Taverna workflow for building the knowledge base with 1152 classification models from the application of 12 classification algorithms on 96 out of 99 datasets. The rest were used for testing. Likewise it was used for extracting data source meta-features.
4. Building of a recommender of algorithms from our data sets with the meta-features chosen by a data mining expert.
5. Evaluation of our recommender in terms of number of times that its answer matches the algorithms that better classify the data set.

In what follows, we describe the datasets and classifiers used in our experiment, along the process of building our knowledge base. Next, we explain the building of our recommender in order to show the feasibility of our proposal.

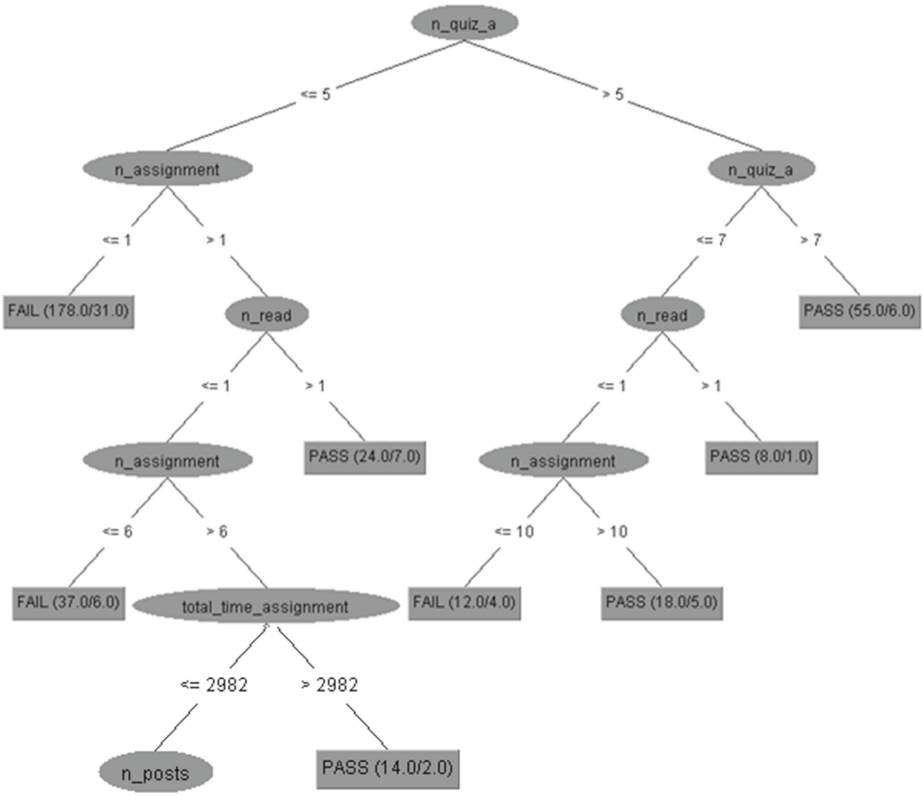

Fig. 7. Graphical view of the pattern obtained from computer science course data.

4.1 Datasets Description

In our experiments, we used data from eight courses hosted in e-learning platforms at University of Cantabria (Spain): (i) one course, entitled "Introduction to multimedia methods" offered in three academic years (2007–2010) with 70 students enrolled in average and hosted in the *Blackboard e-learning platform*; (ii) seven computer science courses taught in the 2007–2008 academic year with a total of 432 enrolled students and hosted in the *Moodle Learning Content Management System*; (iii) six courses oriented to train transversal skills imparted during the first semester of 2013 with a range from 20 to 126 learners per course, also hosted in Moodle; and (iv) a semi-presential course entitled "Mathematics for economists" with 465 students enrolled.

Training Datasets. We defined 23 datasets with information extracted from platforms logs. Each instance in every data set represents the activity of a student in an academic year together with the final mark obtained in the course. Two different groups of datasets are considered: the training data set (used to generate

Table 2. Original datasets description.

Name	# Instances	# Attributes	# numerical Att.	# of nominal Att.	# of classes
data set1	64	13	13	0	2
data set2	65	11	11	0	2
data set3	193	22	22	0	2
data set4	193	22	22	0	4
data set5	193	22	22	0	5
data set6	193	22	0	22	2
data set7	193	22	15	7	2
data set8	64	13	0	13	2
data set9	64	13	7	6	2
data set10	65	11	0	11	2
data set11	65	11	5	6	2
data set12	438	14	14	0	2
data set13	438	14	14	0	4
data set14	438	14	0	14	2
data set15	438	14	5	9	2
data set16	465	6	0	6	2
data set17	465	6	2	4	2
data set18	38	4	0	4	2
data set19	126	5	0	5	2
data set20	28	4	0	4	2
data set21	44	3	0	3	2
data set22	67	6	0	6	2
data set23	67	5	0	5	2

the experiments to feed our knowledge base), and the test datasets (used to evaluate the recommender).

In order to have enough datasets for our experimentation, we generated 96 datasets from them. First we created 3 datasets with data from multimedia course establishing the class attribute with values pass or fail, and another one as the union of these three. The same process was carried out with the programming course, the "Mathematics for economists" course and the transversal courses. Next, we generated 4 discretized datasets from the previous bi-class datasets using PKIDiscretize from Weka, and 4 datasets more but these partially discretized. Besides, we created two datasets with 4 classes (fail, pass, good, excellent) and one with 5 classes (drop-out, fail, pass, good, and excellent). These are our 23 original datasets whose main features are shown in Table 2. Datasets numered from 1 to 11 correspond to the "Introduction to multimedia methods", those from 12 to 15 correspond to the computer science courses, data set 16 and 17 are from the "Mathematics for economists" course and finally datasets numbered from 18 to 23 correspond to the transversal courses.

Table 3. Description of tests datasets.

Name	# instances	# attributes	# numerical att.	# nominal att.	# classes	% missing	class balance
mult2class2010	64	18	18	0	2	0	quite_unbalanced
multGlobalActivity	193	4	4	0	2	0	balanced
tranversalDS	304	6	6	4	2	0	balanced

Then, we generated 72 datasets by adding to the first eighteen from Table 2 a 10 %, 20 %, 30 % and 40 % of missing values. And finally, we created 4 datasets more by applying SMOTE algorithm on 2 of our original datasets with the following proportion of balancing class: 80-20 %, 85-15 %, 70-10 % and 90-10 %.

Test Datasets. Our test datasets are described in Table 3. As can be observed, we chose three datasets with different meta-features: the first one contains the activity carried out by the students in the 2009–2010 academic year in the "Introduction to Multimedia" course (mult2class2010), it is bi-class and all attributes except the class, are numerical; the second one, collects the activity performed in the three editions of Multimedia course degraded with a 10 % of missing values (multGlobalActivity); and finally, the third one gathers data from the six transversal courses mentioned above (tranversalDS) in an unique file. It is bi-class, balanced, without structural nulls, with 2 nominal and 4 numerical attributes. They were used to evaluate the feasibility of our recommender.

4.2 Classifiers Used in the Experiment

Due to the existence of different classification algorithms, 12 different classifiers provided by Weka (trees, rules, bayesian, lazy and ensemble) were introduced into the workflow and executed on the training datasets in order to feed the knowledge base. These classifiers were selected taking into account the most frequently used data mining algorithms [24] and those classifiers used in some previous works about prediction of students performance with which we obtained the best results [7,26]: *J48*, *SimpleCart*, *RandomForest*, *NaiveBayes*, *BayesNetwork*, *Jrip*, *Ridor*, *OneR*, *NNge*, *DecisionTable*, *K-NN*, and *Adaboost*.

4.3 Generating the Knowledge Base

Our knowledge base was fed with results of the training datasets. Each one of the classifiers enumerated in Sect. 4.2 was applied to the 96 training datasets described in Sect. 4.1. Results were stored in the knowledge base, together with their corresponding meta-features described in Sect. 3.3. This means that 1152 different models ($96 * 12$) were generated.

4.4 Results

Before knowing which are the best classifiers for each of the test datasets, we performed a clustering process using kMeans on the meta-features of the training

Table 4. Metadata clustering.

Characteristics	Cluster0	Cluster1	Cluster2	Cluster3	Cluster4
numInstances	438	119.54	512.86	147	401.37
numAtt	14	16.93	14	19	20.11
nominalAtt	85.5	8.68	0	93.29	0
numeicalAtt	15	91.07	100	6.57	100
missingValues	19.62	16.24	12.64	11.19	16.54
is_balanced	QuiteBalanced	Balanced	QuiteBalanced	Balanced	HighlyBalanced

datasets in order to discover if there were well defined patterns that we could remark. In Table 4 we show the results of the 5 clusters obtained. As can be observed, cluster0 collects the datasets with a high number of instances and the nominal attributes and null instances. Cluster1 contains those datasets with the lowest number of instances and a high number of numerical attributes. Cluster2 and cluster4 are very similar, both with a high number of instances and a 100 % of numerical attributes, but differ in the degree of balance, cluster2 gathers quite unbalanced instances and cluster4, highly unbalanced instances. Finally, cluster3 contains instances with a high number of attributes and the highest number of nominal values. This analysis shows that we have a suitable collection of datasets, that means, it is representative enough.

Next, we built classifiers for our test datasets in order to know which one is the technique that best classifies each one. So that, we applied the 12 selected classifiers to the test datasets and these were ranked according to its accuracy. The best algorithms of this ranking are shown in Table 5. The table must be read as follows: the classifier which obtains the best accuracy for the *mult2class2010* data set is *NaiveBayes*, which is followed by *RandomForest* and *NNge*, and quite far by *KNN*, *J48* and *BayesNet*.

Next, we built two different recommenders using *J48* and *NaiveBayes* algorithms, respectively. The meta data set used contained 111 instances, that means, one instance with the meta-features of each data set together the best algorithm which performed the classification task. Since some datasets were classified by more than one algorithm with the same accuracy, these appears twice, once with each algorithm. The data set considered for this task contained the instances of our knowledge base corresponding to the four classifiers that achieved more times the better results, which are (*NaiveBayes*, *J48*, *Jrip* and *BayesNet*).

The recommendation given for each data set by each recommender is shown in Table 6. As can be observed, the recommender based on *J48* recommends, for multGlobalActivity data set, one of the best classifiers, *Jrip*; and the best one for mult2class2012 and transversalDS datasets, *NaiveBayes* and *J48* respectively. The recommender based on*NaiveBayes* recommends one of the best classifiers for the multGlobalActivity data set, *J48*, and for transversalDS data set, *Jrip*. Thus, we conclude that these recommenders select one of the best classification algorithms.

Table 5. Ranking of test datasets when applying classifiers.

Data set	Algorithm	Rank	Accuracy
mult2class2010	NaiveBayes	1	85.9375
	RandomForest	2	82.8125
	NNge	2	82.8125
	kNN	3	79.6874
	J48	4	78.125
	BayesNet	4	78.125
multGlobalActivity	BayesNet	1	84.0206
	SimpleCart	2	83.5052
	DecisionTable	2	83.5052
	J48	3	82.9897
	Jrip	3	82.9897
transversalDS	J48	1	86.1963
	kNN	2	85.5828
	JRip	3	84.3558
	RandomForest	4	83.7423
	SimpleCart	5	83.4653

Table 6. Recommender results.

Data set	J48 Recommendation	NB Recommendation
multGlobalActivity	Jrip	J48
mult2class2010	NaiveBayes	Jrip
transversalDS	J48	Jrip

Finally, we built another recommender, in this case, we used the 12 classifiers described in Sect. 4.2. Results are shown in Table 7. In multGlobalActivity data set, the recommender based on *J48* recommends to use *Jrip*, which is one of the best algorithms to classify this data set. Moreover, for transversalDS data set, it recommends the best classifier, *J48*. The recommender based on *NaiveBayes* also recommends one of the best algorithms for mult2class2010: *RandomForest*. However, the results are worse than in previous experiment in which we only considered four classifiers for our predictive attribute. This happens because, in this case, *RandomForest* appears in knowledge base as the best algorithm in the 25 % of the cases, which is a high percentage over 12 possible classifiers. For transversalsDS data set, it also recommends *RandomForest*, which is the 4th better classifier for this data set over 12.

These results demonstrate that a data mining recommender based in a knowledge base, provides fairly accurate results, and it is cheaper than if evaluated by an expert data miner. Our proposal thus is feasible although it is necessary to have a higher number of experiments in order to get a more general model.

Table 7. Recommender results.

Data set	J48 Recommendation	NB Recommendation
multGlobalActivity	Jrip	RandomForest
mult2class2010	Jrip	RandomForest
transversalDS	J48	Jrip

5 Conclusions and Future Work

The application of data mining techniques are commonly known as a hard process generally based on trial and error empirical methods. As a consequence they can only be applied by a small minority of experts. In this work, a novel approach is defined that (i) uses a taxonomy for identifying user's data mining requirements, (ii) uses a knowledge base which has been defined to store information of data mining experiments with the aim of enabling the building of recommenders that suggest the best algorithm for each data set at hand, and (iii) uses mechanisms to help non-expert data miners to interpret data mining results.

In order to validate that our proposal is feasible, this paper demonstrates that the building of recommenders based on meta-features is highly efficient and effective for our goals. As far as we know we have not found in the literature a similar proposal to ours, focuses on getting outcomes accuracy enough in an easy and economic way. Our proposal thus contributes to advance in the developing of tools oriented to meet the current need of analyzing data by any citizen as stated by [1]. As shown in our experimentation, our approach can be also useful as a resource for other practitioners.

Nevertheless, we must conduct more experiments and also work with more meta-features to enable miners to build more accurate recommenders. Likewise we must develop the end-user workflow and next, studying and analyzing user perception.

Acknowledgments. This work has been partially funded by Open.Mind project (GV/2014/098) from Valencia Government (Spain), by the University Institute for Computing Research (IUII, http://www.iuii.ua.es/) from University of Alicante (Spain), and by the Government of Cantabria under the doctoral studentship program of the University of Cantabria.

References

1. Abadi, D., Agrawal, R., Ailamaki, A., Balazinska, M., Bernstein, P.A., Carey, M.J., Chaudhuri, S., Dean, J., Doan, A., Franklin, M.J., Gehrke, J., Haas, L.M., Halevy, A.Y., Hellerstein, J.M., Ioannidis, Y.E., Jagadish, H., Kossmann, D., Madden, S., Mehrotra, S., Milo, T., Naughton, J.F., Ramakrishnan, R., Markl, V., Olston, C., Ooi, B.C., Christopher, R., Suciu, D., Stonebraker, M., Walter, T., Widom, J.: The beckman report on database research (2013). http://beckman.cs.wisc.edu/beckman-report2013.pdf

2. Blockeel, H., Vanschoren, J.: Experiment databases: towards an improved experimental methodology in machine learning. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 6–17. Springer, Heidelberg (2007). http://dx.doi.org/10.1007/978-3-540-74976-9_5
3. Diamantini, C., Potena, D., Storti, E.: Ontology-driven KDD process composition. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) IDA 2009. LNCS, vol. 5772, pp. 285–296. Springer, Heidelberg (2009)
4. Espinosa, R., García-Saiz, D., Zorrilla, M.E., Zubcoff, J.J., Mazón, J.N.: Development of a knowledge base for enabling non-expert users to apply data mining algorithms. In: Accorsi, R., Ceravolo, P., Cudré-Mauroux, P. (eds.) SIMPDA, CEUR Workshop Proceedings, vol. 1027, pp. 46–61. CEUR-WS.org (2013)
5. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: The kdd process for extracting useful knowledge from volumes of data. Commun. ACM **39**(11), 27–34 (1996)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explorations **11**(1), 10–18 (2009)
7. Hämäläinen, W., Vinni, M.: Comparison of machine learning methods for intelligent tutoring systems. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 525–534. Springer, Heidelberg (2006). doi:10.1007/11774303_52
8. Hilario, M.: e-lico annual report 2010. Université de Geneve, Technical report (2010)
9. Hilario, M., Kalousis, A., Nguyen, P., Woznica, A.: A data mining ontology for algorithm selection and meta-mining. In: ECML/PKDD09 Workshop on Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery, SoKD 2009, pp. 76–87 (2009)
10. Hilario, M., Nguyen, P., Do, H., Woznica, A., Kalousis, A.: Ontology-based meta-mining of knowledge discovery workflows. In: Jankowski, N., Duch, W., Grąbczewski, K. (eds.) Meta-Learning in Computational Intelligence. SCI, vol. 358, pp. 273–315. Springer, Heidelberg (2011)
11. Kalousis, A., Hilario, M.: Model selection via meta-learning: a comparative study. In: 12th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2000, Proceedings, pp. 406–413 (2000)
12. Kietz, J.U., Serban, F., Bernstein, A., Fischer, S.: Designing kdd-workflows via htn-planning. In: Raedt, L.D., Bessière, C., Dubois, D., Doherty, P., Frasconi, P., Heintz, F., Lucas, P.J.F. (eds.) ECAI: Frontiers in Artificial Intelligence and Applications, vol. 242, pp. 1011–1012. IOS Press (2012)
13. Kriegel, H.P., Borgwardt, K.M., Kröger, P., Pryakhin, A., Schubert, M., Zimek, A.: Future trends in data mining. Data Min. Knowl. Discov. **15**(1), 87–97 (2007)
14. Nisbet, R., Elder, J., Miner, G.: Handbook of Statistical Analysis and Data Mining Applications. Academic Press, Boston (2009)
15. Panov, P., Soldatova, L.N., Džeroski, S.: Towards an ontology of data mining investigations. In: Gama, J., Costa, V.S., Jorge, A.M., Brazdil, P.B. (eds.) DS 2009. LNCS, vol. 5808, pp. 257–271. Springer, Heidelberg (2009)
16. Parreiras, F.S., Staab, S., Winter, A.: On marrying ontological and metamodeling technical spaces. In: Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering, ESEC-FSE 2007, pp. 439–448. ACM, New York (2007). http://doi.acm.org/10.1145/1287624.1287687
17. Romero, C., Ventura, S.: Educational data mining: a review of the state-of-the-art. IEEE Tans. Syst. Man and Cybern. Part C Appl. Rev. **40**(6), 601–618 (2010)

18. Romero, C., Ventura, S., García, E.: Data mining in course management systems: moodle case study and tutorial. Comput. Educ. **51**(1), 368–384 (2008). http://dx.doi.org/10.1016/j.compedu.2007.05.016
19. Soldatova, L., King, R.: An ontology of scientific experiments. J. R. Soc. Interface **3**(11), 795–803 (2006)
20. Vanschoren, J., Blockeel, H.: Stand on the shoulders of giants: towards a portal for collaborative experimentation in data mining. In: International Workshop on Third Generation Data Mining at ECML PKDD, 1, 88–89, September 2009
21. Vanschoren, J., Blockeel, H., Pfahringer, B., Holmes, G.: Experiment databases - a new way to share, organize and learn from experiments. Mach. Learn. **87**(2), 127–158 (2012)
22. Vanschoren, J., Soldatova, L.: Exposé: an ontology for data mining experiments. In: International Workshop on Third Generation Data Mining: Towards Service-oriented Knowledge Discovery (SoKD-2010), pp. 31–46, September 2010
23. Vilalta, R., Giraud-Carrier, C.G., Brazdil, P., Soares, C.: Using meta-learning to support data mining. IJCSA **1**(1), 31–45 (2004)
24. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2007). http://dx.doi.org/10.1007/s10115-007-0114-2
25. Záková, M., Kremen, P., Zelezný, F., Lavrac, N.: Automating knowledge discovery workflow composition through ontology-based planning. IEEE Trans. Autom. Sci. Eng. **8**(2), 253–264 (2011)
26. Zorrilla, M.E., García-Saiz, D.: Mining Service to Assist Instructors involved in Virtual Education. Business Intelligence Applications and the Web: Models, Systems and Technologies. Information Science Reference (IGI Global Publishers), September 2011