Development of a knowledge base for enabling non-expert users to apply data mining algorithms

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

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

Abstract. Non-expert users find complex to gain richer insights into the increasingly amount of available 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 data quality of sources. Therefore, these non-expert users must be informed about what data mining techniques and parameters-setting are appropriate for being applied to their sources according to their data quality. To this aim, we propose the construction of an automatic recommender built using a knowledge base which contains information about previously solved data mining tasks. The construction of the knowledge base is a critical step in the recommender design. We propose a model-driven approach for the development of a knowledge base, which is automatically fed by a Taverna workflow. Experiments are conducted to show the feasibility of our knowledge base as a resource in an online educational platform, in which 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, data mining, recommenders, meta-learning, model-driven development

1 Introduction

The increasing availability of data is a great opportunity for everyone to take advantage of their analysis. Physicians in hospitals, lawyers in the law business, teachers in high schools or universities, or even regular citizens, 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 [16].

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 [17, 23] 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 data sets according to their quality. Consequently, 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 what data mining techniques and parameters-setting are appropriate for being applied to their sources according to their data quality.

User-friendly data mining [14] is a step forward to this democratization, since it fosters knowledge discovery without mastering concepts and data mining techniques. To realize user-friendly data mining, in this paper we propose a model-driven approach for the development of a data-mining knowledge base. It contains information about the behavior of data mining algorithms in presence of one or several data quality criteria and intrinsic characteristics of the data sets. This information comes from a set of experiments automatically obtained by means of a Taverna workflow in order to be easily replicated as well as enabling the extension of the knowledge base. A model-driven development approach is proposed in order to obtain the information extracted from our Taverna workflow in a standard manner and automatically generating the knowledge base as a set of models.

The process of building the data-mining knowledge base starts when a dataset is selected as new source in our Taverna workflow. Then, a set of data quality criteria are measured when some mining algorithms are applied to the dataset. This information is stored in a model which is automatically created by using a model-driven approach. An overview of our approach is shown in Figure 1.

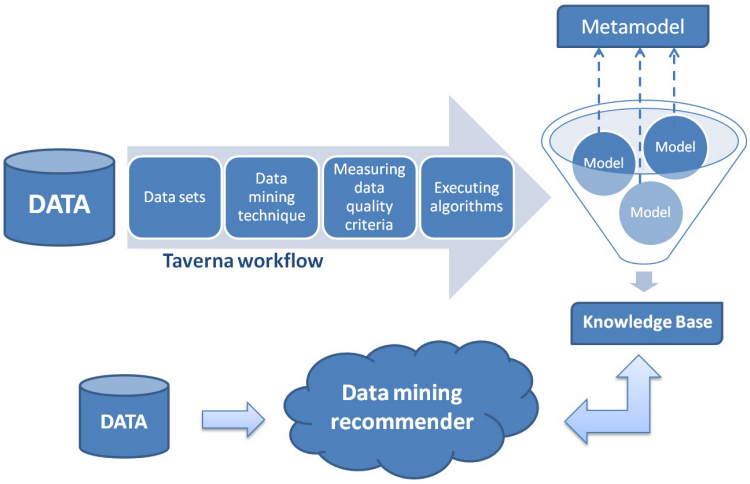

Fig. 1. Developing our knowledge base.

It is worth noting that our knowledge base can be used (i) directly, by non-expert data miners that have certain expertise in data management; or (ii) indirectly, by using a kind of "recommender" that query the knowledge base to guide non-expert data miners by suggesting the best algorithm to be applied to the data, or even to guide experts data miners by suggesting, for example, the algorithms they should use at the beginning of their study.

Some experimentation is conducted in order to evaluate our knowledge base as a resource for a non-expert data miner 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 [21] in the educational context as consequence of the rapid expansion of the use of technologies in supporting learning, not only in established institutional contexts and platforms, but also in the emerging landscape of free, open, social learning online. Although there are tools as ElWM [29] 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 (as non-expert data miners) in applying the right data mining algorithm on their data sets and to extract conclusions oriented to improving the teaching-learning process.

This paper is therefore a step forward to realize the user-friendly data mining. Specifically, the contributions of this paper are as follows:

1. A metamodel that contains those useful concepts for representing models with information about data mining experiments: data's sources metadata, results of data mining algorithms, and values of data quality criteria.
2. A knowledge base as a repository of models that contains the data mining information.
3. A Taverna workflow for providing a mechanism to obtain all the information to automatically create our knowledge base.
4. A set of experiments addressed to build a recommender are shown as proof of feasibility of our approach

The remainder of this paper is structured as follows: an overview of the related work is presented in section 2. Our knowledge base is introduced in section 3, while the conducted experiments are described in section 4. Finally, conclusions and future work are sketched in section 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 [18] is a top-level ontology for data mining concepts that describes basic entities aimed to cover the whole data-mining domain, while EXPO ontology [22] 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 [28] 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 [25] 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 [24]).

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 knowledge base would help naive data miners and non-experts users to have a kind of guidance about which techniques can or should be used and in which contexts.

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 [19]. Under the model-driven umbrella, and according to [13], a model is a "description of (part of) a system written in a well-defined language, while a well-defined language is a language with well-defined form (syntax), and meaning (semantics), which is suitable for automated interpretation by a computer". Therefore, on the one hand, a model must focus on those important parts of a system, thus avoiding superfluous details. On the other hand, well defined languages can be designed by means of metamodeling [1], which provides the foundation for creating models in a meaningful, precise and consistent manner. Therefore, metamodelling 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.

Our aim in this work is creating a metamodel inspired by the aforementioned data mining ontologies that allows us to create a set of models to create a knowledge base for data mining. Moreover, in previous experiments we have demonstrated the influence of data quality in the results obtained when applying techniques of data mining [4].

3 Model-driven approach for knowledge base development

Our knowledge base brings the results on executing data mining processes on many data sets. 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 the data set. Collected information can be useful for supporting non-expert users in a decision making process and which is the best data mining algorithm to apply according to the available data. To this aim, our knowledge base contains the following information:

Information from input data sets. Metadata from the data sets must be known, as number of attributes and instances, as well as the corresponding data types.

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 their results.

Data quality criteria. Several quality criteria from the data sets must be measured. Quality criteria can be related to data sets (e.g. percentages of null values), as well as fields (e.g. field correlation).

3.1 Scientific workflow 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), (ii) creating the knowledge

base, and (iii) implementing a recommender for data mining algorithms based on our knowledge base.

Scientific workflows are largely recognized as useful paradigms to describe, drive, and share information about experiments[5]. Specifically, Taverna Workbench[6] is used in our approach. Taverna is part of myGrid project[7], that aims to produce and use a suite of tools designed to allow international communities to publish and share information.

Our workflow has as a main objective the datasets 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[8]) on which will be applied a set of data mining algorithms. Then, the type of data mining technique must be executed[9]. Next step is about to obtain a predicted attribute (usually the last column). Subsequently, in order to have a visual output in the workflow, expert user can select the resulting algorithm values (e.g. correctly classified instances, mean absolute error, precision, etc.), although 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.

Once required information is acquired, the knowledge base is generated as explained in the following subsection.

3.2 Generating a data mining knowledge base

Our knowledge base aims to represent in a structured and homogeneous manner all the necessary data mining concepts. Following the model-driven paradigm [1], our knowledge base is uniform and automatically created as a repository of models that conforms to a metamodel for representing the output information of our Taverna workflow. Once, the knowledge base is obtained the non-expert miner could use it to evaluate the real dataset in order to obtain the adecuated predicted model having in account the dataset 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 criteria, since the metamodel support creating new quality criteria in each model as required. The definition of our metamodel (see Fig. 3) is based on an analysis of several ontologies (see Section 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 data sets, metadata, data mining algorithms, parameter-setting, data mining results generated when the Taverna workflow is executed, and data quality criteria.

[5] http://en.wikipedia.org/wiki/Scientific_workflow_system
[6] http://www.taverna.org.uk/
[7] http://www.myGrid.org.uk
[8] Attribute-Relation File Format (ARFF), a file format used by the data mining tool Weka [6] to store data.
[9] Our Taverna workflow was designed to be useful for any mining technique, but in this paper we only consider classification techniques.

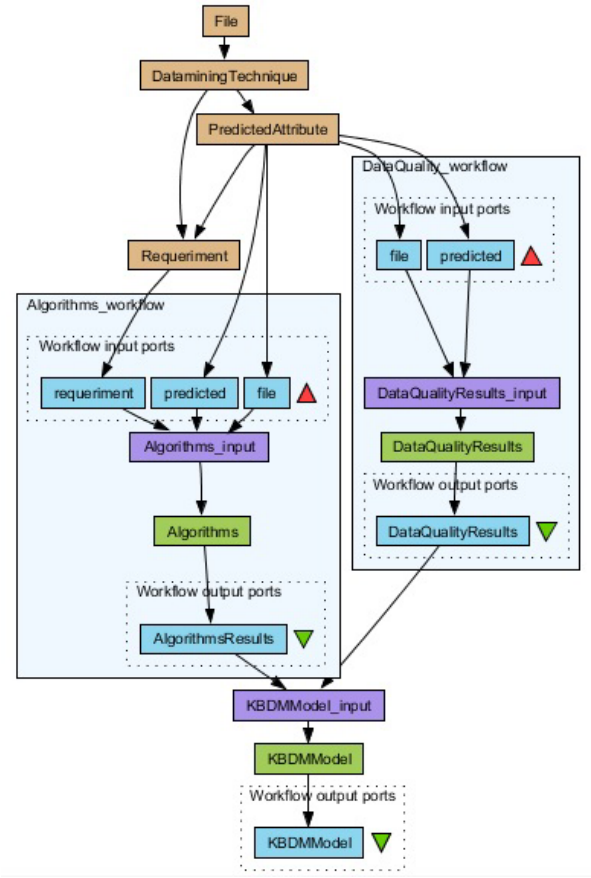

Fig. 2. Our Taverna workflow.

DataSet. It describes data sets 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 (continuous, categorical or mixed).

FieldType. It is an enumeration class for representing the type of each `Field` (numeric, date, nominal or string)

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.

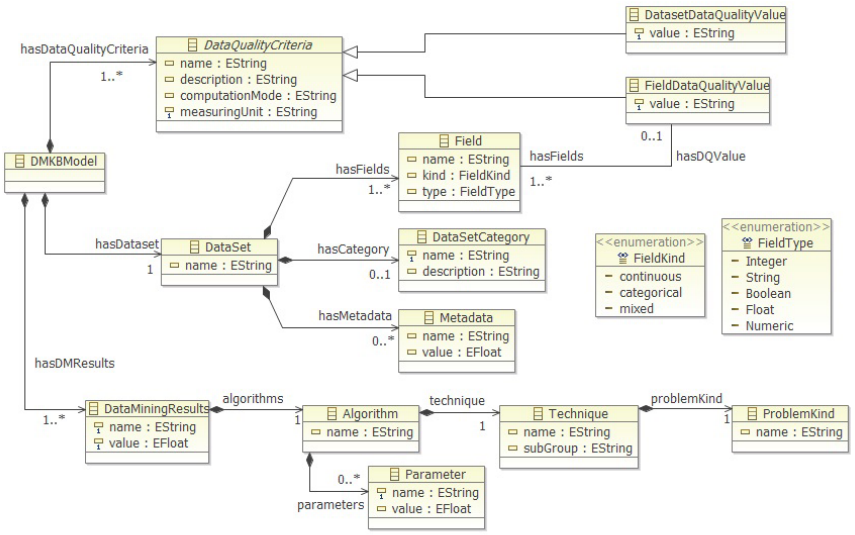

Fig. 3. Our metamodel for representing our data mining knowledge base.

As aforementioned, our Taverna workflow is in charge of handling the model-driven generation of the data mining knowledge base from the acquired information.

When a dataset is processed, all the acquired information is saved in a model conforming to the metamodel presented in Fig. 3. A set of transformations has been developed for creating the models that are integrated in the knowledge base. These transformation are executed in Taverna by means of a Web service.

Our model-driven approach is built on top of the Eclipse Framework[10], which is an open source project conceived as a modular platform which can be extended in order to add features to the development environment. Specifically, transformation tasks for

[10] http://www.eclipse.org

generating models have been supported with the use of Java facilities provided by the Eclipse Modeling Framework (EMF)[11]. The Java code in listing 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`, `technique`, and `problemKind`. Finally, the model (represented by means of a XMI file) is created.

```
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.

Fig. 4 shows a sample `DMKBModel` generated by using our approach. It can be observed some of the elements that conform it (e.g. Dataset, Fields, FieldDataQuality, DatasetDataQuality and DataMiningResults, which refers to the number of correctly classified instances achieved by the Decision Table algorithm for the comp2class dataset, in this case 305).

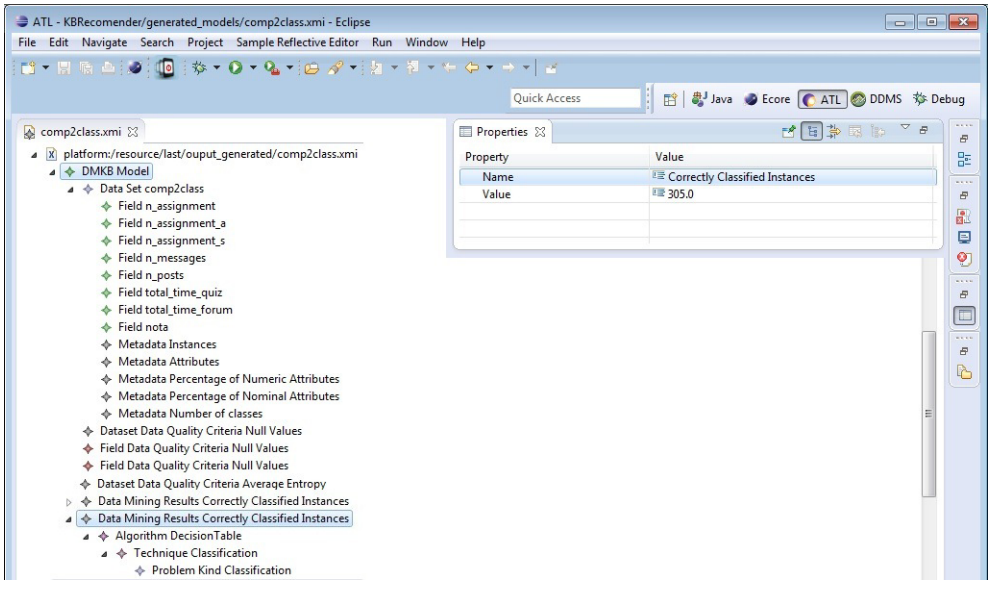

Fig. 4. Sample model of comp2class data set.

[11] http://www.eclipse.org/emf

Our knowledge base is composed by the set of models obtained after running the Taverna workflow for each input data set. These will be the data source which allows us to build our recommender.

3.3 Recommender system

A recommender system takes as input a collection of cases, each belonging to one of a small number of classes and described by its values for a fixed set of attributes, and output a classifier that can accurately predict the class to which a new case belongs (ref [25]). To create this recommender, some different classification algorithms and features can be used. In our case we used as input the number of instances, number of attributes, percentage of nominal attributes, percentage of numerical attributes, percentage of null values, grade of data set balance and algorithm name. We chose these features due to their strong influence in the accuracy which the recommender can achieved.

4 Experimental evaluation

Our approach has been evaluated in the e-learning domain by carrying out a experiment. The methodology followed comprises the steps listed below:

1. Selection of courses and data extraction from e-learning platforms.
2. Generation of 96 data sets as described in Sect. 4.1
3. Building of 1152 classification models from the application of 12 classification algorithms on 96 out of 99 data sets. The rest were used for testing.
4. Extraction of meta-features of each data set
5. Creation of data sets with the meta-features of each data set adding as class attribute the algorithm or algorithms which achieved the highest accuracy.
6. Building of a recommender of algorithms from our data sets with the meta-features chosen. 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, 26].
7. 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 data sets 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.

4.1 Data sets 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 data sets We defined 23 data sets 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 data sets are considered: the training data set (used to generate the experiments to feed our knowledge base), and the test data sets (used to evaluate the recommender).

In order to have enough data sets for our experimentation, and taking into account generally data from virtual learning environments are clean, we built new data sets performing some controlled perturbations to the original datasets. The new data sets have the quality degraded, which allow us to assess if the meta-features chosen are suitable for this purpose. Furthermore, as the process to be performed by the expert should be about a monitored data set which allow validating the behavior of the algorithms under variations of the quality of data.

We generated 96 data sets from them. First we created 3 data sets 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 discretised data sets from the previous bi-class data sets using PKIDiscretize from Weka, and 4 data sets more but these partially discretised. Besides, we created two data sets with 4 classes (fail, pass, good, excellent) and one with 5 classes (drop-out, fail, pass, good, and excellent). These are our 23 original data sets whose main features are shown in Table 1. Data sets 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 data sets numbered from 18 to 23 correspond to the transversal courses.

Then, we generated 72 data sets by adding to the first eighteen data sets from Table 1 a 10, 20, 30 and 40% of missing values. And finally, we created 4 data sets more by applying SMOTE algorithm on 2 of our original data sets with the following proportion of balancing class: 80-20%, 85-15%, 70-10% and 90-10%.

Test data sets Our test data sets are described in Table 2. As can be observed, we chose three data sets 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 knowledge base for building a classifier which helps the end-user in the selection of the best algorithm.

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 executed on the training data sets in order to feed the knowledge base. These classifiers were selected taking into account the most frequently used data mining algorithms [27] and those classifiers used in some previous works about prediction of students performance with

Table 1. Original data sets description

Name	# Instances	# Attributes	# numerical Att.	# of nominal Att.	# of classes
dataset1	64	13	13	0	2
dataset2	65	11	11	0	2
dataset3	193	22	22	0	2
dataset4	193	22	22	0	4
dataset5	193	22	22	0	5
dataset6	193	22	0	22	2
dataset7	193	22	15	7	2
dataset8	64	13	0	13	2
dataset9	64	13	7	6	2
dataset10	65	11	0	11	2
dataset11	65	11	5	6	2
dataset12	438	14	14	0	2
dataset13	438	14	14	0	4
dataset14	438	14	0	14	2
dataset15	438	14	5	9	2
dataset16	465	6	0	6	2
dataset17	465	6	2	4	2
dataset18	38	4	0	4	2
dataset19	126	5	0	5	2
dataset20	28	4	0	4	2
dataset21	44	3	0	3	2
dataset22	67	6	0	6	2
dataset23	67	5	0	5	2

Table 2. Description of tests data sets

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

which we obtained the best results [7, 29]: *J48*, *SimpleCart*, *RandomForest*, *NaiveBayes*, *BayesNetwork*, *Jrip*, *Ridor*, *OneR*, *NNge*, *DecisionTable*, *K-NN*, and *Adaboost*.

4.3 Meta-features

The meta-features used in this experimentation 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.

Next, we explain how was calculated these two last meta-features.

Missing values Structural null values are considered [15]. This kind of null value does not imply that value is not known, but not applicable in certain context. Given that we consider clean our sources of data, given the existence of a null value is considered as a structural null value. The percentage of missing values has been computed by means of $numberofmissingvalues/(numberofattributes * numberofinstances)$.

Balance The unbalanced class criteria is a measure which indicates how unbalanced the class attribute is. Data stored in a certain column are balanced if the numbers of different values representing each different instance are similar, i.e., a similar number of instances are expected for each value. For a two class data set: if balanced of classes is 60-40 or less, then the data set is balanced, else if it is higher than 60-40 but lesser than 80-20, then the data set is quite unbalanced, in other case the data set is highly unbalanced. For multi class data sets (more than 2 classes), the class is highly unbalanced if some of the classes appears more than double than the others. To know how balanced data are, a method that returns the *Chi-square* for each column has been developed. Then, a statistic *Chi-square* test is performed to know if the instances are uniformly distributed. The *null* hypothesis is that all positions have similar number of instances. Then, the data would be uniformly distributed. The alternative hypothesis states that they are different. The level of significance (the point at which one can determine with 95% of confidence that the difference is not due to chance alone) is set at 0.05. The *Chi-square* formula is as follows:

$$\chi^2_{obs} = \sum_{i=1}^{n} \frac{(f_i - np_i)^2}{np_i}$$

where

- χ^2_{obs}:
- f_i: number of observed frequencies.
- p_i: number of expected frequencies.
- n is the number of categories to be considered.

4.4 Generating the knowledge base

Our knowledge base was fed with results of the training data sets. Each one of the classifiers enumerated in Section 4.2 was applied to the 96 training data sets described in Section 4.1. Results were stored in the knowledge base, together with their corresponding meta-features described in Section 4.3. This means that 1152 different models ($96 * 12$) were generated.

4.5 Results

The knowledge base is used by a recommender for selecting 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 supports the recommender in its task, and on the other hand, evaluating the goodness of our recommender.

Before knowing which are the best classifiers for each of the test data sets, we performed a clustering process using kMeans on the meta-features of the training data sets in order to discover if there were well defined patterns that we could remark. In table 3 we show the results of the 5 clusters obtained. As can be observed, cluster0 collects the data sets with a high number of instances and the nominal attributes and null instances. Cluster1 contains those data sets 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 data sets, that means, it is representative enough.

Table 3. 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

Next, we built classifiers for our test data sets 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 data sets and these were ranked according to its accuracy. The best algorithms of this ranking are shown in Table 4. 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*.

Table 4. Ranking of test data sets 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	823.7423
	SimpleCart	5	83.4653

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 data sets 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 5. 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 dataset, *J48*, and for transversalDS data set, *Jrip*. Thus, we conclude that these recommenders select one of the best classification algorithms.

Table 5. Recommender results.

Dataset	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 Section 4.2. Results are shown in Table 6. 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.

Table 6. Recommender results

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

These results demonstrate that our proposal is feasible although it is necessary to have a higher number of experiments in order to get a more general model. It is a little problem in e-learning context because although there are lots of courses hosted in e-learning platforms, not all courses can be used since it is necessary to know how the courses were designed and exploded by learners to be considered to predict performance.

We used other techniques based on landmarking [20] but the results were worse. On the other hand, we should add other meta-features related to parameter-setting of the algorithms. In this experimentation the algorithms were run with their default parameters.

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 paper, a knowledge base is defined that contains information of previous data mining experiments in order to provide guidance to non-expert users to apply data mining techniques. To generate our knowledge base, a model-driven approach is defined, based on a Taverna workflow. As shown in our experiments, our knowledge base can be useful as a resource for non-expert data miners. The best classifiers can be recommended most of times from a set of 4 classifiers (*NaiveBayes*, *J48*, *Jrip*, and *BayesNet*) in order to predict the performance of students in our e-learning scenario. Moreover, in one of these cases, our knowledge base supports in recommending the best algorithm for two of the data sets. Although, the number of good recommendations were worse when the set of classifiers is 12, these results encourage us to continue researching in order to improve how the recommender can use the knowledge base in a better manner. As future work, we plan to conduct more experiments in order to study how to obtain better results when more classifiers are considered. Regardless the recommender can provide good results to a non-expert user with significantly low effort, more complex recommenders that improve these results could be developed.

Acknowledgments. This work has been partially funded by IN.MIND project from University of Alicante (Spain).

References

1. Bézivin, J.: On the unification power of models. Software and System Modeling 4(2), 171–188 (2005)
2. Blockeel, H., Vanschoren, J.: Experiment databases: Towards an improved experimental methodology in machine learning. In: Kok, J., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenic, D., Skowron, A. (eds.) Knowledge Discovery in Databases: PKDD 2007, Lecture Notes in Computer Science, vol. 4702, pp. 6–17. Springer Berlin / Heidelberg (2007), http://dx.doi.org/10.1007/978-3-540-74976-9_5, 10.1007/978-3-540-74976-9_5
3. Diamantini, C., Potena, D., Storti, E.: Ontology-driven kdd process composition. In: IDA. pp. 285–296 (2009)
4. Espinosa, R., Zubcoff, J.J., Mazón, J.N.: A set of experiments to consider data quality criteria in classification techniques for data mining. In: ICCSA (2). pp. 680–694 (2011)
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., Chan, T.W. (eds.) Intelligent Tutoring Systems. Lecture Notes in Computer Science, vol. 4053, pp. 525–534. Springer Berlin / Heidelberg (2006), 10.1007/11774303_52
8. Hilario, M.: e-lico annual report 2010. Tech. rep., Université de Geneve (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. pp. 76–87. SoKD-09 (2009)
10. Hilario, M., Nguyen, P., Do, H., Woznica, A., Kalousis, A.: Ontology-based meta-mining of knowledge discovery workflows. In: Meta-Learning in Computational Intelligence, pp. 273–315 (2011)
11. Kalousis, A., Hilario, M.: Model selection via meta-learning: a comparative study. In: Tools with Artificial Intelligence, 2000. ICTAI 2000. Proceedings. 12th IEEE International Conference on. 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. Kleppe, A., Warmer, J., Bast, W.: MDA Explained. The Practice and Promise of The Model Driven Architecture. Addison Wesley (2003)
14. 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)
15. Mazón, J.N., Lechtenbörger, J., Trujillo, J.: A survey on summarizability issues in multidimensional modeling. Data Knowl. Eng. 68(12), 1452–1469 (2009)
16. Mazón, J.N., Zubcoff, J.J., Garrigós, I., Espinosa, R., Rodríguez, R.: Open business intelligence: on the importance of data quality awareness in user-friendly data mining. In: EDBT/ICDT Workshops. pp. 144–147 (2012)
17. Nisbet, R., Elder, J., Miner, G.: Handbook of Statistical Analysis and Data Mining Applications. Academic Press (2009)
18. Panov, P., Soldatova, L.N., Dzeroski, S.: Towards an ontology of data mining investigations. In: Discovery Science. pp. 257–271 (2009)
19. 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. pp. 439–448. ESEC-FSE '07, ACM, New York, NY, USA (2007), http://doi.acm.org/10.1145/1287624.1287687
20. Pfahringer, B., Bensusan, H., Giraud-carrier, C.: Meta-learning by landmarking various learning algorithms. In: Proceedings of the 17th International Conference on Machine Learning. pp. 743–750 (2000)
21. Romero, C., Ventura, S.: Educational Data Mining: A Review of the State-of-the-Art. IEEE Tansactions on Systems, Man and Cybernetics, part C: Applications and Reviews 40(6), 601–618 (2010)
22. Soldatova, L., King, R.: An ontology of scientific experiments. J R Soc Interface 3(11), 795–803 (2006)
23. Vanschoren, J., Blockeel, H.: Stand on the Shoulders of Giants: Towards a Portal for Collaborative Experimentation in Data Mining. International Workshop on Third Generation Data Mining at ECML PKDD 1, 88–89 (Sep 2009)
24. Vanschoren, J., Blockeel, H., Pfahringer, B., Holmes, G.: Experiment databases - a new way to share, organize and learn from experiments. Machine Learning 87(2), 127–158 (2012)
25. 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 (Sep 2010)
26. Vilalta, R., Giraud-Carrier, C.G., Brazdil, P., Soares, C.: Using meta-learning to support data mining. IJCSA 1(1), 31–45 (2004)
27. Wu, X., Kumar, V., Ross Quinlan, J., 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 (Dec 2007), http://dx.doi.org/10.1007/s10115-007-0114-2
28. Záková, M., Kremen, P., Zelezný, F., Lavrac, N.: Automating knowledge discovery workflow composition through ontology-based planning. IEEE T. Automation Science and Engineering 8(2), 253–264 (2011)
29. Zorrilla, M.E., García-Saiz, D.: Business Intelligence Applications and the Web: Models, Systems and Technologies, chap. Mining Service to Assist Instructors involved in Virtual Education. Information Science Reference (IGI Global Publishers) (September 2011)