“Semantics Inside!” But let’s not tell the Data Miners: Intelligent Support for Data Mining

Jörg-Uwe Kietz,[1] Floarea Serban,[1] Simon Fischer,[2] and Abraham Bernstein[1]

[1]DDIS, University of Zurich, Switzerland*
[2]Rapid-I, Dortmund, Germany
{serban,juk,bernstein}@ifi.uzh.ch; fischer@rapid-i.com

Abstract. Knowledge Discovery in Databases (KDD) has evolved significantly over the past years and reached a mature stage offering plenty of operators to solve complex data analysis tasks. User support for building data analysis workflows, however, has not progressed sufficiently: the large number of operators currently available in KDD systems and interactions between these operators complicates successful data analysis. To help Data Miners we enhanced one of the most used open source data mining tools—RapidMiner—with semantic technologies. Specifically, we first annotated all elements involved in the Data Mining (DM) process—the data, the operators, models, data mining tasks, and KDD workflows—semantically using our eProPlan modelling tool that allows to describe operators and build a task/method decomposition grammar to specify the desired workflows embedded in an ontology. Second, we enhanced RapidMiner to employ these semantic annotations to actively support data analysts. Third, we built an Intelligent Discovery Assistant, eIda, that leverages the semantic annotation as well as HTN planning to automatically support KDD process generation.
We found that the use of Semantic Web approaches and technologies in the KDD domain helped us to lower the barrier to data analysis. We also found that using a generic ontology editor overwhelmed KDD-centric users. We, therefore, provided them with problem-centric extensions to Protégé. Last and most surprising, we found that our semantic modeling of the KDD domain served as a rapid prototyping approach for several hard-coded improvements of RapidMiner, namely correctness checking of workflows and quick-fixes, reinforcing the finding that even a little semantic modeling can go a long way in improving the understanding of a domain even for domain experts.
Keywords: #eswc2014Kietz

1 Introduction

Over the last years Knowledge Discovery for Large Databases (KDD) has grown immensely and attracted more focus from both research and industry since large

* The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2011 under grant agreement No 231519.

amounts of data have been generated that need to be analyzed. New algorithms and methods have been proposed and even integrated in current Data Mining (DM) tools. But KDD is a complex process transforming the raw data into actionable knowledge by applying a series of successive algorithms. Current DM tools allow users to manually build KDD workflows and select each step from a large pool of possible solutions. Yet, this is very tedious and often leads to workflows that crash after a few hours runtime. In addition, it has been shown that users and even DM experts tend to stick to a set of preferred methods and do not explore the entire design-space [13]. This often produces sub-optimal analyses. Despite the progress such tools have made during the last years, their user support is, hence, still far from perfect.

Some of the issues highlighted above have been explored by the EU funded e-Lico project.[1] An important task of this project was to *provide intelligent support for DM to simplify the data analysis process for users.* Inspired by [4] the plan was to intelligently support users by leveraging the semantic annotation of DM-operations to allow a correct composition of workflows. Hence, we annotated the main components of KDD workflows (data, algorithms, and goals) and operators available in RapidMiner [14]—one of the most used open source Data Mining tools—using ontologies. Treating the problem as a service composition problem, where algorithms are services provided by DM tools, we employed ontologies, Hierarchical Task Network (HTN) planning [7] to automatically explore the large design-space of correct workflows and recommender-system technology to help users navigate this space [20].

As KDD is a dynamic domain, we developed eProPlan—a tool that enables DM service providers to efficiently model their services and define semantics—facilitating the semantic description process and providing tools for testing and maintaining the model over time. eProPlan was used by several partners (including the RapidMiner developers) to model different operators (e.g., basic DM operators, text mining operators, etc.). As Section 5 outlines, insights from this modeling effort where included in the productive RapidMiner version to check for correctness of workflows and suggest possible fixes.We also designed an Intelligent Discovery Assistant (IDA), eIda, that enhances RapidMiner and Taverna [16] with the ability to automatically generate semantically correct workflows (i.e., exploring the design space of valid workflows) starting only from a dataset and a DM problem. Finally, we developed a recommender-component that can rank generated workflows in terms of their expected accuracy, which is able to advise a user on which of the workflows generated by eIda are expected to perform well [20] simplifying the choice significantly.

This paper's main contributions are the distillation of our experiences implementing the e-Lico tool-suite, the evaluations we undertook of the different parts, and the lessons we learned from applying semantic technologies in this increasingly important application area that is so heavily influenced by induction instead of logical reasoning. In particular, it is witness how insights from

[1] `http://www.e-lico.eu/`

modeling the KDD domain transcended into the actual code of one of the most used KDD-software.[2]

The discussion is structured as follows: Next, we succinctly explore related work, followed by a discussion of the three implemented systems: eProPlan for describing the semantics of services in Section 3, eIda to automatically generate the KDD workflows in Section 4, and the auto-experimenting recommender in Section 4.3. The side effects of semantics are discussed in Section 5. Finally we explore the lessons learned from this project in Section 6 and we draw the main conclusions in Section 7.

2 Related Work

Researchers have been extensively looking at service composition especially in the area of web services [5,15]. The most common automation techniques for web services are either based on workflow composition [3] or AI planning [18]. Some even used a more advanced form of AI planning called Hierarchical Task Network (HTN) to find the matching web services [24]. Similarly, we use HTN planning to generate the design-space of KDD workflows. In contrast to others, we did not look for *one possible* solution but were exploring *the space of all correct solutions*. Also we did not rely on OWL-S but simple ontologies, where the conditions and effects were stored as annotations.

Other researchers looked into automating KDD [21]. Most of those tools are, however, limited to being research prototypes and did not provide explicit support for modeling their background knowledge. The Intelligent Discovery Automatic Assistant (IDEA) [4] was among the first prototypes to propose a combination of ontology-based planning, result ranking, and operator information sharing for supporting KDD. The RDM system [26] uses an OWL-DL [17] ontology for knowledge discovery. This ontology is queried using the Pellet reasoner [23] and SPARQL-DL queries [22] for retrieving the operator inputs and outputs, which are then fed into the planner. Two AI planners are used that query the ontology during the planning process.Similarly, KDDVM [6] interacts with the ontology by using the Pellet reasoner and SPARQL-DL queries. Instead of applying a standard planning algorithm it utilizes a custom algorithm that starts at the goal state and iteratively adds operators forming a directed graph until the first operator is compatible with the given dataset. Operators are added using an algorithm matching procedure, which checks the compatibility of inputs and outputs. The operator interfaces are not matched perfectly do not need to be perfectly but according to a similarity computed via their distance in the ontology graph. Finally, the produced workflows are ranked based on the similarity scores as well as other operator properties stored in the ontology (e.g. soft pre-conditions or computational complexity estimates). Finally, MLWizard,[3]

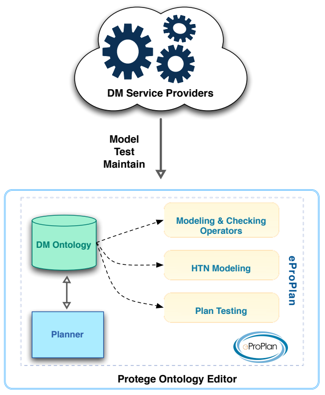
[2] According to a recent poll `http://www.kdnuggets.com/2013/06/kdnuggets-annual-software-poll-rapidminer-r-vie-for-first-place.html`

[3] `http://madm.dfki.de/rapidminer/mlwizard`

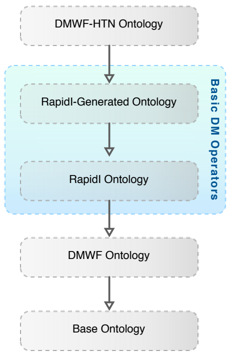
e.g., an IDA available at the RapidMiner marketplace only supports a few classification tools and no regression and clustering. It also focuses the induction step and does not consider evaluation (e.g. building a cross-validation or test-set evaluation process) and preprocessing (e.g. handling missing values, attribute type conversions, and normalization).

3 eProPlan: A Semantic DM Service IDE

To reason and plan about KDD operators one needs to specify the main element of the KDD domain and the capabilities of each individual operator (or service) in terms of their IOPE (Inputs, Outputs, Pre-conditions, and Effects). To facilitate these tasks we built eProPlan[4] [10] to be used by the DM service providers who are the DM experts. They have the knowledge about the services, how their services work, what data they can model, what they produce, and how they are implemented. eProPlan is a Protégé plugin that not only allows to describe semantically the services but also to improve, test, debug, and extend the service model as well as the HTN used for planning (see Figure 1a). In the following we first discuss eProPlan's ontology and models before discussing its features supporting editors and its usage.



(a) eProPlan components



(b) Ontologies

Fig. 1: eProPlan system and its ontologies

3.1 DM Ontology for Workflow Planning

In our case services are actual algorithms that can be organized by their capabilities and data applicability. Hence, one can use an ontology to structure and model them. We built a set of OWL-DL ontologies that are describing the DM domain. To ensure decoupling and re-use we built different ontologies organized in a stack (see Figure 1b), where each ontology imports the one directly below it (and the others below it indirectly).

[4] which is available from the e-Lico web-site.

The Base ontology (see Figs 1b and 2) contains the building blocks for modeling the services (operators). *Operators* use different inputs and produce one or several outputs—all *IO-Object*s. They also have different types: abstract (used only to organize operators in hierarchies), basic (can be executed), and dominating or loops (allowing a sequence of operators that can be repeated). Parameters are used to tune algorithms or to set mandatory values by defining *data-* or *object properties* (e.g., *parameter, simpleParameter*).

The composition of services starts by defining a *Goal* that takes as input the data that needs to be analyzed. Next, the main goal uses a *Task* that can be solved by one or more *Methods*. Each method has then a set of steps that represent the services in the composition. Such a step can either be a *Task* or an *Operator* allowing the definition of very complex workflows and even loops over the same tasks.

The main advantages of embedding the HTN-grammar into an ontology are that (1) operators are organized in an hierarchy and (2) abstract operators can be used in the HTN-grammar. The abstract HTN operator *ClassificationLearner*, e.g., has several executable sub-concepts in RapidMiner and Weka. This enabled a compact and, often, execution-system independent HTN-grammar.

The DM Workflow ontology (DMWF in Figs 1b and 2) defines sub-classes for all the basic classes in the Base ontology. The focus of this ontology is on the composition of specific services from DM. Operators are specified for each step of the workflow from pre-processing to evaluation (e.g., *FormatData, Modeling, Model-Evaluation, etc.*). Data can be of different types as for example *DataTable, Model* or *Report*. It has nominal or ordinal columns and attributes. The characteristics of data are specified using sub-classes of the *MetaData* class (e.g., *Attribute, AttributeType, DataColumn, DataFormat*, etc.). The *MainGoal* materializes now to concrete tasks from DM like *Descriptive-* or *PredictiveModeling*.

Here we leveraged the completion of *IO-Object*'s via ABox realization and SWRL-rule reasoning, which enabled compact and understandable descriptions similar to axioms in the Planning Domain Description Language (PDDL) [25]. We, e.g., defined a concept for missing value (MV) free data tables $MVFreeDataTable \equiv DataTable \sqcap \forall inputColumn.MVFreeColumn \sqcap \forall targetColumn.MVFreeColumn$, where $MVFreeColumn$ in itself is a defined concept $MVFreeColumn \equiv DataColumn \sqcap amountOfMVs = 0$

The RapidI-Generated ontology (see Figure 1b) specifies the RapidMiner provided algorithms (services). The ontology contains the RapidMiner operators as well as their pre-conditions and effects. These are rules that define when certain operators can be used in terms of the characteristics of the data and what they produce (either new data or changes on the input data). OWL-S [1] already provides the semantics and models for web service composition. Its support for editing in Protégé was limited when we built eProPlan and was difficult to explain to our user-base (KDD domain experts and RapidMiner developers) – experiences that were reflected by others [19]. Also, it modeled services as instances. To simplify matters we chose to describe services as OWL classes. To

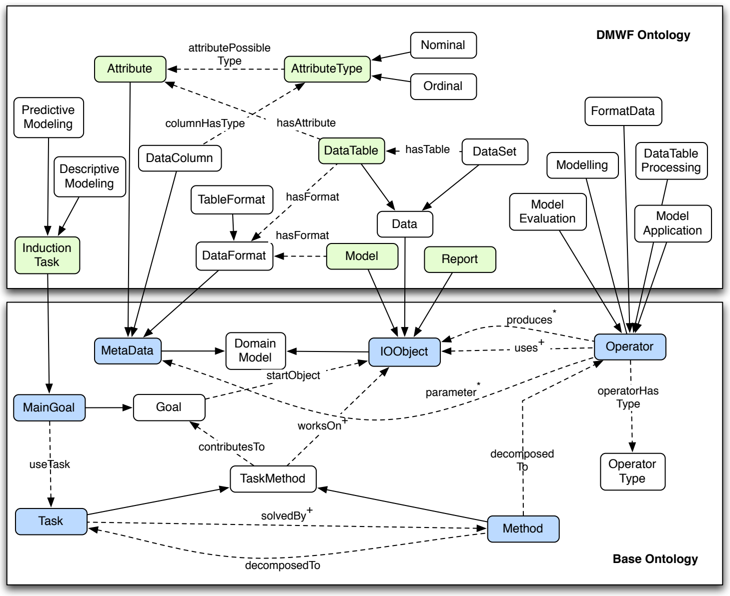


Fig. 2: Main classes from the Base and Workflow ontologies

model the pre-conditions and effects we relied on the Protégé's SWRL plugin. Since the complexity of the rules needed was higher than it was possible to express in SWRL, we have added some new built-ins [12] and saved them as annotations in the ontology.

The operators are organized in an strict inheritance concept-hierarchy. We used the OWL-inheritance to propagate parameter restrictions from abstract to concrete, implemented operators as well as conditions/effect representations. This provided an effective way to build and maintain descriptions of over hundred different operators (see section 2.3 in[11] for an example).

The DMWF-HTN ontology (see Figure 1b) is used to define the tasks and methods for planning. HTN planning is like a top-down grammar that specifies the skeleton or template of workflows. Each step of the workflow can be described in term of specific tasks or even sub-tasks (e.g., *DataMining, ModelingWithCrossValidation, Preprocessing, CleanMissingValues, CleanManyValuedNominals, NormalizeScalar*, etc.). Further, the planner employs the template and fills in the services that match. The planning relies on external reasoners available in Protégé for TBox reasoning and an internal ABox reasoner. Details of the operator models and of the used HTN-planning can be found in [11]

Domain Independence The modularity of our approach would allow using our tools in another application domain by starting with the Base ontology and then adapting the upper elements. To ensure that this actually works we also modeled the Missionaries and Cannibals problem from classical planning. Hence, eProPlan can be used as a generic ontology editor for HTN planning for different domains.

3.2 eProPlan: Support for editing, testing, and optimization

To allow domain experts to model operators/services for planning we developed eProPlan, which has several Protégé views. The *Operators tab* allows users to define conditions and effects and includes a semantic checker that highlights errors in the definition and suggests fixes. This avoids erroneous definitions of services and maintains ontology correctness.

The operator's applicability can be tested using the *Applicable operators tab*, where each operator can be applied on different types of data. Furthermore, it supports a step by step composition to ensure that correct workflows can be generated. Since standard OWL-reasoners do not support states, we built the planner in Flora2/XSB.[5] Consequently, the ontological domain needs to be compiled for planning at which stage the conditions and effects are checked again for correctness and the errors are stored as annotations to the corresponding operators. This ensures that the planning domain is defined correctly.

In the *HTN editor tab* one can create new tasks and methods and specify each step of the workflow. The editor shows the hierarchy of tasks and methods and also each independent step with their inputs and outputs. The user needs to define the matching IO-Objects from one step to another to ensure the correctness of the workflow.

3.3 Usage of eProPlan

eProPlan was used and tested by different parties as it is the main tool developed for the e-Lico project. The partners from Rapid-I were responsible for modeling the RapidMiner operators. Without prior experience in ontology modeling they successfully delivered a complex ontology that contained more than 100 modeled operators. As the number of services is relatively large (a few hundreds) they automatically generated the code by mapping their algorithm specifications to the conditions and effects in the ontology. Some additional work was then needed to check the correctness of the produced service model using eProPlan.

To gather feedback about the usability of eProPlan we designed a questionnaire based on the System Usability Scale (SUS) [2]. SUS has been used in many tests and is often described as a "quick and dirty" usability test. SUS scores can range from 0 (very little satisfaction) to 100 (very high satisfaction). Usual average satisfaction scores are reported to be between 65 and 70. For eProPlan we obtained SUS questionnaires from 5 users varying from 42.5 to 72.5 with a mean score of 63.5. [8] reports a SUS score for Protégé (with 15 subjects) that varies between 20 to 78 with a mean of 47.[6] We are well aware that comparing SUS scores between settings is highly problematic. We can, therefore, only infer that eProPlan's usability seems comparable to Protégé's.

[5] `http://flora.sourceforge.net/`

[6] That is rather a rather low score for Protégé might be the result of its comparative evaluation setting with with CLOnE, a Controlled Language Ontology Editor.

4 eIDA – a tool for users

eIDA is a programming interface that provides all the functionality (in particular to the reasoner & planner) needed to build an Intelligent Discovery Assistant (IDA). It provides methods for retrieving the DM workflows starting from the dataset's meta-data and the selection of a main goal. So far it was used to build IDAs for both RapidMiner and Taverna.[7] The RapidMiner IDA Extension can be downloaded (or even auto-installed) from the Rapid-I Marketplace [8]. So far it was downloaded over 8000 times. The Taverna extension[9] can execute the workflows generated by the IDA[10] using any RapidAnalytics[11] server that provides all RapidMiner operators as web-services. Extensions for other KDD tools (e.g., KNIME, Enterprise Miner, etc.) would require two steps: first modeling their corresponding operators in the DMWF, second an implementation of the GUI and the plan-converter using the IDA-API.

We tested the IDA on 164 datasets from the UCI repository of Machine Learning datasets. [12] It produced executable plans for all 117 classification and 47 regression problems. These datasets have between 3 and 1558 attributes, being all nominal (from binary to many different values like ZIP), all scalar (normalized or not), or mixed. They also have varying degrees of missing values. We are not aware of any other Machine Learning or DM approach that is able to adapt itself to so many different and divergent datasets. The IDA also works for less well prepared datasets like the KDD Cup 1998 challenge data (370 attributes, with up to 50% missing values and nominal data), where it generates plans of around 40 operators. Generating and ranking 20 of these workflows took 400 sec. on a 3.2 GHz Quad-Core Intel Xeon.

4.1 Ease of Use

Without an IDA data mining is typically done by specialized highly-trained professionals such as DM consultants. They have to know a lot about DM methods and how they are implemented in different tools. They need to inspect the data and combine the operators into an adequate workflow. However, the large number of available algorithms is overwhelming even for specialists.

The IDA reduces the technical burden and offers "DM with 7 clicks" (see Figure 3). (1) Show the IDA-Perspective of the tool; (2) drag the data to be analyzed from the repository to the view or import (and annotate) your data; (3) select your main goal in DM; (4) ask the IDA to generate workflows for data and goal; (5) evaluate all plans by executing them in RapidMiner; (6) select the plan you like most to see a summary of the plan (the screenshot in Figure 3 is

[7] `http://www.taverna.org.uk/`

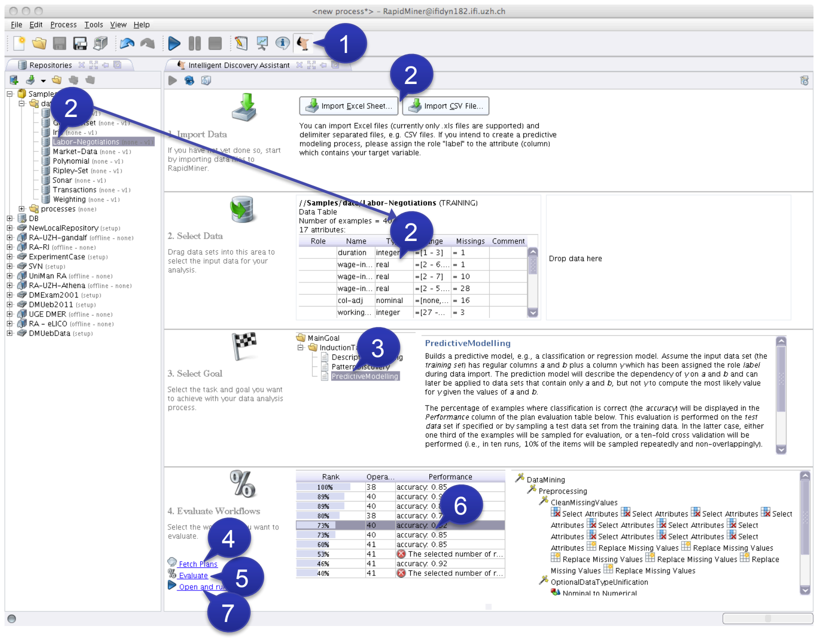
[8] `http://rapidupdate.de/UpdateServer/faces/product_details.xhtml?productId=rmx_ida`

[9] `http://e-lico.eu/taverna-ida.html`

[10] `http://e-lico.eu/taverna-rm.html`

[11] `http://rapid-i.com/content/view/182/196/`

[12] `http://archive.ics.uci.edu/ml/`



<new process*> – RapidMiner@ifidyn182.ifi.uzh.ch
File Edit Process Tools View Help
Repositories
Intelligent Discovery Assistant
Import Excel Sheet...
Import CSV File...
1
2
2
2
3
4
5
6
7
Drop data here
Fetch Plans
Evaluate
Open and run

1. Import Data

If you have not yet done so, start by importing data files to RapidMiner.

You can import Excel files (currently only .xls files are supported) and delimiter separated files, e.g. CSV files. If you intend to create a predictive modeling process, please assign the role "label" to the attribute (column) which contains your target variable.

2. Select Data

Drag data sets into this area to select the input data for your analysis.

//Samples/data/Labor-Negotiations (TRAINING)
Data Table
Number of examples = 40
17 attributes:

Role	Name	Type	Range	Missings	Comment
	duration	integer	=[1 - 3]	= 1	
	wage-in...	real	=[2 - 6....	= 1	
	wage-in...	real	=[2 - 7]	= 10	
	wage-in...	real	=[2 - 5....	= 28	
	col-adj	nominal	=[none,...	= 16	
	working...	integer	=[27 -...	= 3	

3. Select Goal

Select the task and goal you want to achieve with your data analysis process.

MainGoal
InductionTask
DescriptiveModelling
PatternDiscovery
PredictiveModelling

PredictiveModelling

Builds a predictive model, e.g., a classification or regression model. Assume the input data set (the *training set*) has regular columns *a* and *b* plus a column *y* which has been assigned the role *label* during data import. The prediction model will describe the dependency of *y* on *a* and *b* and can later be applied to data sets that contain only *a* and *b*, but not *y* to compute the most likely value for *y* given the values of *a* and *b*.

The percentage of examples where classification is correct (the *accuracy*) will be displayed in the *Performance* column of the plan evaluation table below. This evaluation is performed on the *test data* set if specified or by sampling a test data set from the training data. In the latter case, either one third of the examples will be sampled for evaluation, or a ten-fold cross validation will be performed (i.e., in ten runs, 10% of the items will be sampled repeatedly and non-overlappingly).

4. Evaluate Workflows

Select the workflows you want to evaluate.

Rank	Opera...	Performance
100%	38	accuracy: 0.85
89%	40	accuracy: 0.9
89%	40	accuracy: 0.8
80%	38	accuracy: 0.7
73%	40	accuracy: 0.92
73%	40	accuracy: 0.85
60%	41	accuracy: 0.85
53%	41	The selected number of r...
46%	41	accuracy: 0.92
40%	41	The selected number of r...

DataMining
Preprocessing
CleanMissingValues
Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Select Attributes Replace Missing Values Replace Missing Values Replace Missing Values Replace Missing Values Replace Missing Values Replace Missing Values
OptionalDataTypeUnification
Nominal to Numerical

Fig. 3: IDA Interface in RapidMiner

made after this step); and finally, (7) inspect the plan and its results. Note that these steps do not require any detailed technical knowledge. Still a user should be aware of what (s)he is doing when (s)he uses DM, i.e. (s)he should know the statistical assumptions underlying DM (e.g., a user should know what it means to have a sample that is representative, relevant, and large enough to solve a problem with DM/statistics).

4.2 Speedup of Workflow Design

Besides making DM easier for inexperienced users, our main goal in building the IDA was to speed-up the design of DM workflows. To establish a possible speed-up we compared the efficiency of CS students after attending a DM class to a person using the IDA. The study comprises in total 24 students (9 in 2011 and 15 in 2012). They had to solve the following DM problems:

– Take the UCI "Communities and Crime" data-set[13] and
  - **a)** generate a fine clustering that allows to find very similar communities
  - **b)** generate a description of the clusters (learn a model to predict the cluster label built in task a)).
  - **c)** generate a function to predict "ViolentCrimesPerPop" and evaluate it with 10-fold cross-validation.

[13] `http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime`

– Take the UCI "Internet Advertisement" data-set[14] and generate an evaluated classification for the attribute "Ad/Non-Ad".

All data was provided already imported into RapidMiner (via a local RapidAnalytics server they could access). All students needed the full 3 hours (see Figure 10 in [11] for details on typical errors the students made and accuracy's reached by IDA and students)

As a comparison we asked a non-specialist (a member from our Group, not involved in the e-LICO project) to accomplish the same task using the IDA. It took the non-specialist 30 minutes to (IDA planning, minimal manual adaptation, and execution of the workflows) with a comparable output.

The study confirmed that standard DM problems (such as clustering and prediction tasks on complex UCI data) can be sped-up considerably (30 minutes vs. 3 hours) by using an IDA whilst maintaining a comparable quality
The experiments also showed clearly that even students at the end of a term-long DM class were overwhelmed when confronted by two typical DM task. The IDA operated by a DM novice, with minimal guidance about which proposed workflow to use,[15] was able to provide comparable results when. Note that the students are an optimal user-group for the IDA, as they have limited DM experience but they understand the principles of DM.

4.3 Auto-experimentation and Performance of the generated workflows

One of the most complex aspects of Data Analysis is to decide which KDD workflows are likely to be successful. Hence, we employed eIDA for another research task: the ranking of the generated KDD workflows. This is an important aspect since the number of valid workflows is quite large (easily over 1000 for classification problems) due to the large number of operators available in RapidMiner and modeled in our ontology. To solve this problem we combined auto-experimentation (the planing and execution of experiments) with collaborative filtering [20]. The detailed description of this approach is clearly beyond the scope of this paper. We summarize that we evaluated the ranking on 100 datasets from the UCI repository. For new datasets our approach was able to make recommendations that were at most 5% worse than the best workflow by executing only 3%-5% of the overall workflows.

This excellent result shows that *an IDA based on Semantic Technology and collaborative filtering approaches is able to automatically decide how to analyze a data-set with a result that is close to the best possible analysis*. We believe that this result clearly shows the power of Semantic Technology applied to DM problems.

[14] `http://archive.ics.uci.edu/ml/datasets/Internet+Advertisements`
[15] a problem we addressed with auto-experimentation (discussed in the next section)

5 Side Effects of Semantic Technology Usage

Besides the development of the systems described in this manuscript, several improvements were made as side effects in the core of the RapidMiner user interface. Those simple things in fact contributed a lot to the usability of RapidMiner as a data analysis workbench. They were one of the most relevant features culminating in the release of RapidMiner 5. We will briefly sketch them in this section.

RapidMiner improvements When building the DM ontology as described in Section 3.1, a lot of work went into the specification of pre-conditions and effects of operators. This information is not only useful for the DM ontology and the IDA, but also when designing processes in the traditional way. The RapidMiner-team therefore, decided to make as much information as possible also available in Java. To that end, as of RapidMiner 5.0, each operator can (and all operators in the core do) provide so-called meta-data propagation rules. They must be very quick, since they are frequently evaluated at process design time, whenever changes to the process are made. They serve three purposes.

First, they compute, given the (possibly incompletely specified) meta data delivered at the input ports, what the meta data of the output will be. Note that sometimes, this information cannot be known at design time, but only at run-time. A good example is the *Pivot* operator which transforms data values into new attributes. Since the set of all values is not necessarily known without loading the full data set, the complete set of attributes after the execution of the *Pivot* operator may not be known before execution time. Another example is loading data from dynamic sources like Web services. In such cases, the information can be marked as partially unknown.

Second, once the meta data are evaluated, these rules check pre-conditions required by the individual operators. As an example, a learning scheme like an SVM requires all input attributes to be numerical. If these pre-conditions are not satisfied, a warning is displayed in a special view (akin to errors in IDEs) and the operator is highlighted in red to signify that it cannot be executed. This saves the user the effort of test-wise executing the process for debugging.

Finally, when an operator is found to be non-applicable quick-fixes are offered to the user. Using the example above, when the user tries to apply an SVM on non-numerical data, a quick fix could be to insert an operator for dummy-coding nominal values (as shown in Figure 4).

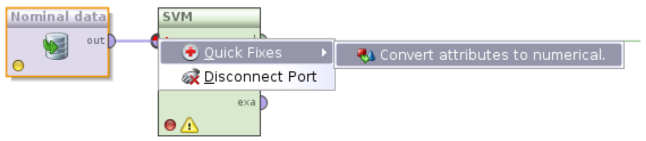


Fig. 4: A quickfix for making an SVM applicable on non-numeric data

In summary, the semantic annotation of the DM operators done for the IDA have served as a rapid-prototype that inspired the further development of RapidMiner. It significantly improved the usability of RapidMiner, which is supported by a significant increase of users and community activity after the release of RapidMiner 5 (including over 350'000 downloads and an increase in market-share from 21%[16] to 37.8%[17] since its launch).

Sharing scientific workflows Designing KDD workflows is a complex task requiring both a good understanding of the problem and of the algorithms available in KDD tools. This makes sharing of such workflows a valuable feature akin to sharing program code: it allows researchers to convey their knowledge and experience to others. Imagine that people can browse through a repository of good performing workflows for a certain area and then they can test and evaluate them on their datasets. This is especially useful for novice data miners who do not know all the subtleties of the domain and do not have the required experience to avoid mistakes. But it may be beneficial even for specialists, as they may come across interesting workflows that they did not use or think about before.

Another side-effect of our modeling of the DM domain is that sharing these workflows has become easier. Within e-Lico we developed a myExperiment [9] plugin for RapidMiner. It allows scientists to have groups per research topics and share various workflows and their experience on different data. Whenever people get good workflows using the RapidMiner IDA Extension or when they manually design them, they can now share them using the myExperiment plugin. Both Taverna and RapidMiner offer this feature. Free re-usability, search, and storage of workflows is, therefore, ensured. myExperiment users can tag, write reviews, comments, and even annotate and rate workflows.

The e-Lico project participants have already shared and uploaded several dozen DM workflows on this platform, and several hundreds have been added since then by the community.

6 Lessons learned

During the 3 year EU project we were facing the challenge of adapting and intermingling various Semantic Web techniques to build an IDA for the data analysis process. This required us to support two types of users and, therefore, provide different types of support: the DM service providers who are responsible for modeling the DM domain and the end-users who "only" want to magically get valuable knowledge out of their dataset/task at hand. An important lesson discovered in the first year is that *service providers are not ontology modeling specialists and, therefore, need debugging support for Semantic Modeling*, despite being AI specialists. In addition, *end-users are not interested in semantic technology* and the underlying details. They want to solve problems, in our case

[16] http://www.kdnuggets.com/polls/2009/data-mining-tools-used.htm

[17] http://www.kdnuggets.com/polls/2010/data-mining-analytics-tools.html

KDD analyses, with as little effort as possible. *The less we showed them about the inner workings (and semantic technology) the happier they were.*

Ontologies have become more popular and are used for modeling and structuring different domains. Ontology editors have adapted and provide different plugins to facilitate the transition to the semantic notations. From our experience with the users from the project we observed that domain specialists are extremely reluctant to climb the learning curve to use these editors. Therefore, to simplify and expedite their learning process, we found it helpful to build domain specific ontology editors (such as eProPlan) that hide the basic ontology constructs by replacing them with domain constructs. These facilitate bridging the application domain and semantic annotation. Reinforcing this finding we asked all e-Lico project participants what their most liked eProPlan feature was. The winner was the special editor for conditions and effects as well as the applicable operators' tab–both elements highly applicable to the DM operator modeling problem they had to solve. They also appreciate having any kind of validation and especially a way of finding out the problems or errors in their modeling. Consequently, it seems that *one of the biggest hurdle for wide-spread Semantic annotation is the provision of low-effort domain-specific editors with built-in validation facilities.*

One of the biggest findings of our project is that *the payoff of unexpected side-effects of using semantic technology may easily surpass the effort invested.* In the spirit of "a little semantics goes a long way"[18] we could enhance RapidMiner's interface with simple semantic functions that immensely simplified its usage. Whilst we lack proof that this is the cause of the surge of RapidMiner's popularity we gathered evidence from user testing at Rapid-I that a number of test users were able to fix problems with the quick-fix functionality that specialist data miners did not think were possible. Even if the semantic techniques where not used in this improvements, they served as a rapid-prototyping approach that influenced the further development of RapidMiner.

Last but not least, *the ability to explore the design-space of a realistic-sized DM domain and successfully propose well-performing workflows replied on semantic technology and allowed us to solve practical problems that have been pursued for decades* [21].

7 Conclusions

In this paper we presented our experiences with using semantic technologies in the KDD/DM domain. In this domain the number of available operators and resulting design space of possible DM workflows goes beyond the capability of even specialists. Our solution couples HTN planning as well as auto-experimentation with recommendations with semantic web technologies to address this problem.

Specifically, we provided eProPlan, an editor and planning environment for services, and eIda, an Intelligent Discovery Assistant engine that was included in

[18] `http://www.cs.rpi.edu/~hendler/LittleSemanticsWeb.html`

both RapidMiner and Taverna. Leveraging the functionalities of these tools we were able to semantically annotate the KDD domain and use those annotations in practical usage: we found that these tools relying on semantic technologies were able to support novice users to get good results in data mining tasks. To put it to real-world tests we shipped these capabilities with RapidMiner. Furthermore, we found that when complementing these tools with advanced auto-experimentation based recommendation technology we could automatically propose good performing workflows.

We also found that the side-effects of Semantic annotation led to a series of high-impact improvements in the usability of RapidMiner indicating that "a little semantics goes a long way" – in our case helping to boost the popularity of RapidMiner. Most interestingly, *we managed to put semantic technology at work on the computers of tens of thousands and influenced the user experience of possibly hundreds of thousands users making their work simpler whilst leaving them completely oblivious about the usage of such technology* – a goal that we believe every semantic technology project should strive for.

References

1. A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, et al. Daml-s: Web service description for the semantic web. In *International Semantic Web Conference (ISWC)*, pages 348–363. Springer, 2002.
2. A. Bangor, P. Kortum, and J. Miller. Determining what individual SUS scores mean: adding an adjective rating scale. *Journal of Usability Studies*, 4(3):114–123, 2009.
3. B. Benatallah, M. Dumas, M.-C. Fauvet, and F. A. Rabhi. *Towards patterns of web services composition.* Springer, 2003.
4. A. Bernstein, F. J. Provost, and S. Hill. Toward intelligent assistance for a data mining process: An ontology-based approach for cost-sensitive classification. *IEEE Trans. Knowl. Data Eng.*, 17(4):503–518, 2005.
5. D. B. Claro, P. Albers, and J.-K. Hao. Web services composition. In *Semantic Web Services, Processes and Applications*, pages 195–225. Springer, 2006.
6. C. Diamantini, D. Potena, and E. Storti. Kddonto: An ontology for discovery and composition of kdd algorithms. In *Service-oriented Knowledge Discovery (SoKD 2009) Workshop at ECML/PKDD 2009*, pages 13–24, 2009.
7. K. Erol. *Hierarchical task network planning: formalization, analysis, and implementation.* PhD thesis, University of Maryland at College Park, College Park, MD, USA, 1996. UMI Order No. GAX96-22054.
8. A. Funk, V. Tablan, K. Bontcheva, H. Cunningham, B. Davis, and S. Handschuh. Clone: Controlled language for ontology editing. In K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web, Proc. of the 6th ISWC and 2nd ASWC*, number 4825 in LNCS, pages 142–155. Springer, 2007.
9. C. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, and D. De Roure. myExperiment: a repository and social network for the sharing of bioinformatics workflows. *Nucl. Acids Res.*, 2010.

10. J. Kietz, F. Serban, and A. Bernstein. eProPlan: A Tool to Model Automatic Generation of Data Mining Workflows. In *3rd Planning to Learn Workshop at ECAI 2010*, page 15, 2010.
11. J. Kietz, F. Serban, and A. Bernstein. Designing kdd-workflows via htn-planning. In J. Vanschoren, P. Brazdil, and J.-U. Kietz, editors, *4rd Planning to Learn Workshop at ECAI 2012*, volume Vol-950 of *CEUR Workshop Proceedings*, 2012.
12. J. Kietz, F. Serban, A. Bernstein, and S. Fischer. Towards cooperative planning of data mining workflows. In *Proceedings of the ECML-PKDD'09 Workshop on Service-oriented Knowledge Discovery*, pages 1–12, 2009.
13. R. Kohavi, C. E. Brodley, B. Frasca, L. Mason, and Z. Zheng. Kdd-cup 2000 organizers' report: peeling the onion. *SIGKDD Explor. Newsl.*, 2:86–93, December 2000.
14. I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale: Rapid prototyping for complex data mining tasks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940. ACM, 2006.
15. N. Milanovic and M. Malek. Current solutions for web service composition. *Internet Computing, IEEE*, 8(6):51–59, 2004.
16. T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, et al. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, 2004.
17. P. Patel-Schneider, P. Hayes, I. Horrocks, et al. OWL web ontology language semantics and abstract syntax. *W3C recommendation*, 10, 2004.
18. J. Rao and X. Su. A survey of automated web service composition methods. In *Semantic Web Services and Web Process Composition*, pages 43–54. Springer, 2005.
19. M. Sabou, D. Richards, and S. Van Splunter. An experience report on using daml-s. In *The Proceedings of the Twelfth International World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW'03). Budapest*, 2003.
20. F. Serban. *Towards Effective Support for Data Mining using Intelligent Discovery Assistance*. PhD thesis, University of Zurich, Department of Informatics, 2013.
21. F. Serban, J. Vanschoren, J.-U. Kietz, and A. Bernstein. A survey of intelligent assistants for data analysis. *ACM Computing Surveys*, forthcoming:Epub ahead of print–Epub ahead of print, 2013.
22. E. Sirin and B. Parsia. SPARQL-DL: SPARQL query for OWL-DL. In *Proceedings of the International Workshop on OWL Experiences and Directions (OWLED)*, 2007.
23. E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.
24. E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau. Htn planning for web service composition using shop2. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(4):377–396, 2004.
25. S. Thiébaux, J. Hoffmann, and B. Nebel. In defense of pddl axioms. *Artif. Intell.*, 168(1):38–69, Oct. 2005.
26. M. Žáková, P. Křemen, F. Železný, and N. Lavrač. Automatic knowledge discovery workflow composition through ontology-based planning. *IEEE Transactions on Automation Science and Engineering*, online 1st:53–264, 2010.