DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at SEGI UNIVERSITY or other institutions.

Signature : ________________________________

Name : ________________________________

ID No. : ________________________________

Data : ________________________________

APPROVAL FOR SUBMISSION

I certify that this project report entitled "WEB SERVICE FRAMEWORK FOR MINING EDUCATIONAL DATA" was prepared by FELERMINO DÁRIO MÁRIO ANTÓNIO ALI has met the required standard for submission in partial fulfilment of the requirements for the award of Master of Science in Information Technology at SEGi University.

Approved by,

Signature :____

Supervisor : Dr. Ashley Ng Sok Choo

Date :____

COPYRIGHT

ACKNOWLEDGEMENTS

First of all, my gratitude to God Almighty for his protection. I would like to express my heartfelt thankfulness to Dr. Ashley Ng Sok Choo and Dr. Leelavathi Rajamanickam for their enthusiastic supervisions. They provided me with insights that guided me and challenged my thinking, substantially improving the thesis. Also, I would like to convey my personal and deepest appreciation to my lecturers and IGS staff, for all the assistance rendered to me during my study at SEGi University.

Special thanks are due to Luis Filipe, Realdo Dias, and Fidel Bilika, the most helpful friends that I have ever had. They have made my studies enjoyable.

All of whom have helped me tremendously throughout the course of my studies, I express my deepest gratitude. Finally, the study was in part supported by SEGi University Scholarship and Lúrio University.

WEB SERVICE FRAMEWORK FOR MINING EDUCATIONAL DATA

ABSTRACT

E-learning is an evolutionary form of distance learning and has become better over time as new technologies emerged. Today, efforts are still being made to embrace E-learning systems with emerging technologies in order to make them better. Among these advancements, Educational Data Mining (EDM) is one that is gaining a huge and increasingly popularity due to its wide application for improving the teaching-learning process in online practices. However, even though EDM promises to bring many benefits to educational industry in general and E-learning environments in particular, its principal drawback is the lack of easy to use tools. The current EDM tools usually require users to have some additional technical expertise to effectively perform EDM tasks. Thus, in response to this limitations, this study intends to design and implement an EDM application framework which aims at automating and simplify the development of EDM in E-learning environment. The application framework introduces a Service-Oriented Architecture (SOA) that hides the complexity of technical details and enables users to perform EDM in an automated fashion. The framework was designed based on abstraction, extensibility, and interoperability principles. The framework implementation was made up of three major modules. The first module provides an abstraction for data gathering, which was done by extending Moodle LMS (Learning Management System) source code. The second module provides data mining methods and techniques as services, it was done by converting Weka API into a set of Web services. The third module acts as an intermediary between the first two modules, it contains a user-friendly interface that allows to dynamically locate data provider services, and run knowledge discovery tasks on data mining services. An experiment was conducted to evaluate the overhead of the proposed framework through a combination of simulation and implementation. The experiments shown that the overhead introduced by the SOA mechanism is relatively small, therefore, it has been concluded that a service-oriented architecture can be effectively used to facilitate educational data mining on E-learning environments.

TABLE OF CONTENTS

LIST ABBREVIATIONS AND NOTATIONS

AEHS	Adaptative Educational Hypermedia System
AI	Artificial Intelligence
ARFF	Attribute-Relation File Format
CSCL	Computer Supported Collaborative Learning
CSV	Command Separated Value
DDM	Distributed Data Mining
DM	Data Mining
DS	Design Science
EDM	Educational Data Mining
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
ITS	Intelligent Tutoring Systems
KDD	Knowledge-Discovery in Databases
KT	Knowledge Tracing
LAN	Local Area Network
LMS	Learning Management System

MDPWS	Moodle Data Provider Web Service
NMF	Non-negative Matrix Factorization
OGSA	Open Grid Services Architecture
REST	REpresentational State Transfer
SMTP	SOAP over Simple Mail Transfer Protocol
SNA	Network Analysis
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SOEDM	Service-Oriented Educational Data Mining
UUDI	Universal Description, Discovery, Integration specification
WDMWS	Weka Data Mining Web Service
WSDL	Web Services Definition Language
WSRF	Web Services Resource Framework
XML	eXtensible Markup Language

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

1. Introduction

1.1. Background of study

For a long time, traditional classroom instruction has been the main prevail model of education. But, over the years and with the technological advancements, a new type of instruction based on information and communication technologies (ICTs) has emerged. E-learning, Web-based learning, online learning, computer-based education and distance learning are widely used as interchangeable terms to refer to this new paradigm of instruction. It uses online technologies as an intermediary platform to connect educators and learners separated by time or space (Umadevi, Maheswari, & Nithya, 2014). In fact, E-learning has initially begun in the form of correspondence (regular transfer of printed materials), then through multi-media, next depending on information technology (e.g. audio-video conferences and radio transmission), and nowadays through the internet (HEEPF, 2016). Since then, advancement never stopped, new improvements have been continuously emerging, and E-learning systems became better. One of the efforts being made at the present time is the integration Data Mining (DM) into E-learning systems, which is also known as Educational Data Mining (EDM). EDM aims at extracting useful information out of the data accumulated over the educational systems (e.g. traditional classrooms, E-learning, adaptive and intelligent web-based systems), and use this information to improve user’s experiences within the educational system.

Figure 1.1 gives an illustration of the EDM cycle in educational systems. Taking E-learning systems as an example, the process of applying EDM starts, for instance, by collecting user's usage data, which might be the information related to user interaction with E-learning's resources (i.e., forums, chats, emails, assignments, quizzes, and archives). Next, some DM algorithms are applied to the collected data, resulting in the creation of a particular model. Then, once this model is created, follows the interpretation and the assessment of the model and ultimately the utilization of the model for enhancement of E-learning user's experience.

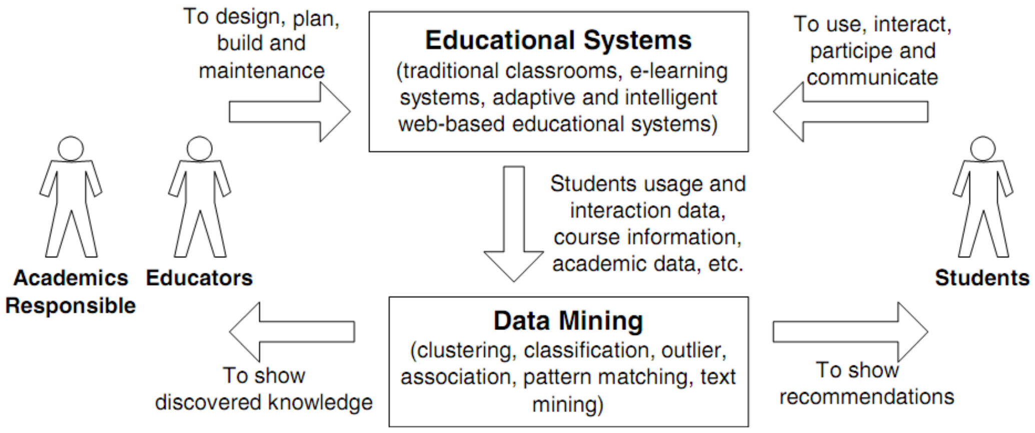

To design, plan, build and maintenance
Educational Systems
(traditional classrooms, e-learning systems, adaptive and intelligent web-based educational systems)
To use, interact, participe and communicate
Students usage and interaction data, course information, academic data, etc.
Academics Responsible
Educators
Students
To show discovered knowledge
Data Mining
(clustering, classification, outlier, association, pattern matching, text mining)
To show recommendations

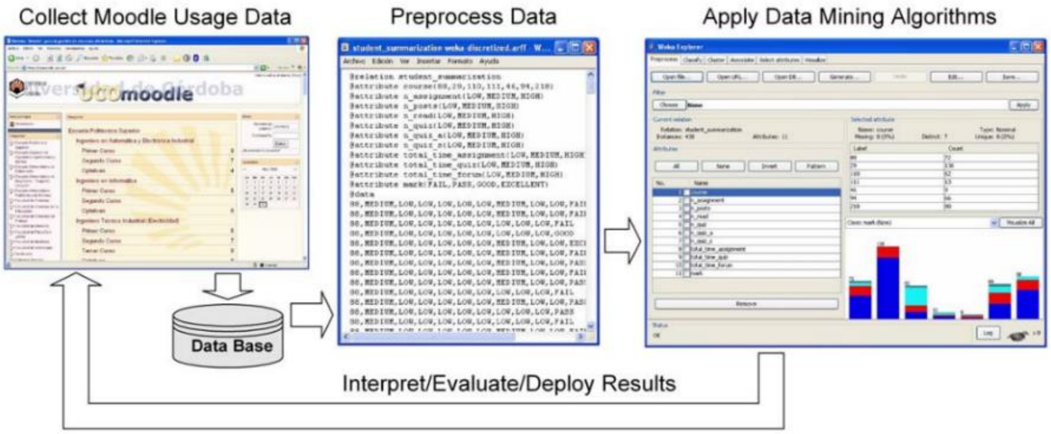
Figure 1.1: The cycle of applying EDM in educational system (C. Romero & Ventura, 2007)

Over the last few years, it has been reported great impacts of EDM in educational researches, especially, in the improving of the teaching and learning practices in E-learning environments (C. Romero & Ventura, 2010). E-learning environments because EDM methods can be used to resolve some of the shortcomings caused by the lack of face-to-face interaction between educators and learners in E-learning systems. For instance, EDM has been commonly used by the EDM research community to predict of students' performance in E-learning systems, also to identify and explain usual and unusual learning behaviours in E-learning systems. These methods allowed educators and students in an E-learning system to take an in-time intervention in order to improve their teaching and learning practices, hence, preventing failures or underachievement in online courses.

However, even though EDM promises to bring many benefits to educational industry in general and E-learning environments in particular, its principal drawback is the lack of easy to use tools. The current existing EDM tools are too complex, and usually require users to have some additional technical expertise to effectively perform EDM tasks. Thus, this study attempts to build an EDM tool for mining E-learning's data so that users can perform EDM in E-learning in an automated fashion.

1.2. Problem statement

In E-learning environments, EDM is a three steps process which includes collecting and pre-processing data, then apply EDM algorithms for extracting hidden patterns from the data, and finally, the interpretation and evaluation of the results obtained (see Figure 1.2).


Collect Moodle Usage Data
Universidad de Córdoba
UCOmoodle
Data Base
Preprocess Data
student_summarization weka discretized.arff - W...
Archivo Edición Ver Insertar Formato Ayuda
Apply Data Mining Algorithms
Weka Explorer
Interpret/Evaluate/Deploy Results

```
@relation student_summarization
@attribute course{88,29,110,111,46,94,218}
@attribute n_assignment{LOW,MEDIUM,HIGH}
@attribute n_posts{LOW,MEDIUM,HIGH}
@attribute n_read{LOW,MEDIUM,HIGH}
@attribute n_quiz{LOW,MEDIUM,HIGH}
@attribute n_quiz_a{LOW,MEDIUM,HIGH}
@attribute n_quiz_s{LOW,MEDIUM,HIGH}
@attribute total_time_assignment{LOW,MEDIUM,HIGH
@attribute total_time_quiz{LOW,MEDIUM,HIGH}
@attribute total_time_forum{LOW,MEDIUM,HIGH}
@attribute mark{FAIL,PASS,GOOD,EXCELLENT}
@data
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,FAI
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,FAI
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,LOW,LOW,LOW,FAIL
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,LOW,LOW,LOW,GOOD
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,EXC
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,FAI
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,PAS
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,FAI
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,PAS
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,LOW,LOW,LOW,FAIL
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,MEDIUM,LOW,LOW,PAS
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,LOW,LOW,LOW,PASS
88,MEDIUM,LOW,LOW,LOW,LOW,LOW,LOW,LOW,LOW,FAIL
```

Figure 1.2: EDM In E-learning environments (Cristóbal Romero, Ventura, & García, 2008)

Each of these steps can be very challenging, especially, for educational users who are not expert in DM such as a student or an educator. Therefore, many tools have been proposed by the EDM research community in order to facilitate the process of applying EDM in educational contexts. In general, these tools can be grouped into two categories: general purpose tools and specialized tools for EDM. One of the problems with using the general purpose tools is that they all run on a single machine. Moreover, they can be inadequate for solving some specific educational problems, and typically, the process of preparing the data in order to be used in these tools takes time, requires huge efforts and consume lots of resources. For this reasons, many researchers in EDM have demanded the creation of specialized tools for EDM, in order to simplify the development of EDM researches and practices. As a response to this call, the EDM research community has proposed several tools specialized for EDM, where, a majority of them were applied on E-learning environments since E-learning environments provide supportive tools (digital data, programmable, etc.) that enable the automation of some EDM tasks.

However, even with the efforts made in order to bring specialized tools for EDM, the currently existing tools are still failing to meet some of the most requested requirements by the EDM research community such as reusability, extensibility, usability, and maintainability (Cristóbal Romero, Romero, Ventura, & Data, 2014). In addition, the majority of the existing EDM tools are just prototypes, most of which provide restricted features, tend to fulfil one research goal, and above all, they do not meet all EDM community requirements (Nabila Bousbia & Belamri, 2014; Cristóbal Romero et al., 2014; Cristobal Romero & Ventura, 2013). As an example, one of the inconvenience of the currently existing tool is that they usually require that the users have to select the tables that they are interested in, apply the right filters and configurations for modelling the DM tasks, as well as understand enough of the DM techniques to be able to make a choice amongst methods as to which method is the most suitable in the context, and even how to interpret the results (DeFreitas & Bernard, 2014).

Therefore, this research attempts to improve the EDM tools for E-learning environments, by designing an EDM application framework using distributed knowledge discovery that enables to hide the complexity of some EDM technical details so that users can perform EDM in an automated fashion. The use of distributed knowledge discovery technology appears as a solution to the pointed limitations because it provides important features such as extensibility, aggregation, scalability, resilience and the ability to hide implementation details, and hence provide the basis for a well-architected distributed knowledge discovery applications.

1.3. Research objective

The main objective of this research is to improve the EDM tools for E-learning environments, by designing an EDM application framework using distributed knowledge discovery that enables to hide the complexity of some EDM technical details, so that users can perform EDM in an automated fashion. To achieve the objective, following tasks are fulfilled:

1. Identify the key requirements of EDM tools for E-learning;
2. Investigate the use of the distributed knowledge discovery as a mechanism for automation of EDM tasks in E-learning systems;
3. Design and develop an easy-to-use EDM application framework;
4. Identify comprehensible and accurate algorithms for predicting student’s performance;
5. Develop a Moodle module for predicting student’s performance based on the students' usage data.
6. Assess the performance of the proposed framework.

1.4. Research questions

1. What are the requirements of an EDM tool for E-learning environments?
2. How to design an EDM tool that will enable to hide the complexity of some EDM technical details, so that users can perform EDM in an automated fashion in E-learning environments?
3. What accurate and comprehensible model can be used for predicting student's final marks based on the students' usage data?
4. What are the execution times of different steps needed to perform a typical EDM task in different LAN (Local Area Network) scenarios?
5. What is the efficiency of the distributed knowledge discovery mechanism to automate and simplify the development of EDM in E-learning environments?

1.5. Significance of Study

This research proposes an EDM application framework which applies distributed knowledge discovery as a mechanism for automation of EDM tasks in E-learning systems. The contributions of this study are:

- Reduce significantly the time, effort and resource needed to perform EDM in E-learning environments;
- Reduce the level of effort among developers and allowing them to focus on a specific module of interest rather than requiring the redevelopment of the entire application architecture for every research question within educational data mining;
- Allow future developers to add new features such as algorithms as well as E-learning data sources without need to change the overall structure of the application;

- Standardization of input data and output model, therefore, allowing to deal with multiple E-learning environments as well as facilitate the creation of easy-to-understand user interface for non-technical users such as educators and students.

1.6. Scope of study

In this thesis, an EDM application framework is proposed. It is a system with a modular architecture composed of three major module: Data Provider, DM Service, and integration module. The first prototype developed in this research only include libraries that only supports connection with Moodle LMS and Weka DM framework. Allowing the remote execution of pre-processing, prediction, clustering and association rule mining methods. In addition, this study proposed a report on Moodle system that allows educators to predict students' performance based on models generated by the EDM application framework proposed in this study.

1.7. Thesis organization

This thesis contains six chapters, including this introductory chapter which covers the background information as well as introduces the problem statement, the objectives and the achievements of this research. The rest of the thesis is organized in different chapters as follows: Chapter 2 – involves a critical and comprehensive review of the literature toward to answer of the research objectives. Chapter 3 – describes in details the proposed framework and its implementation. Chapter 4 – describes the experimental methodology which is subject to the research questions of this study. Chapter 5 – discusses and reports the results obtained from experimental data. Finally, the thesis concludes with Chapter 6 which states the major conclusion of this research and explores promising directions for future work and development.

CHAPTER 2

2. Literature review

2.1. Introduction

This chapter gives a bibliographic review of the different studies made in the field of educational data mining (EDM) and the most relevant studies. The aim is to seek answers to the following questions:

- What is EDM? What is the relationship between EDM and education? What are the objective of EDM, the process, the methods and techniques for EDM?
- How people perform EDM in E-learning? What tools do they use? What barriers do they face?
- What distributed knowledge discovery mechanism can be used for solving the problem of EDM tools? Why?

This chapter is organised into two section, addressing the questions outline above: Educational Data Mining and Distributed knowledge discovery.

2.2. Educational Data Mining

Today's digital evolution has created a society that is highly dependent on technology. Nowadays, there are many applications out there to assist people in almost every aspect of day-to-day life, from a simple task as managing a daily to-do list, to complex tasks as driving a car. Under this circumstance, a huge amount of data is being generated in almost every single minute and stored in massive databases. Therefore, the need to create methods and techniques to uncover hidden patterns and get insights from this large collection of data, has led to the birth of Data Mining (DM), also known as Knowledge-Discovery in Databases (KDD).

By definition, DM is an interdisciplinary field that combines statistics, machine learning, artificial intelligence and databases technology, to analyse data from different perspectives and summarizes the result as meaningful information. DM is a versatile tool, it can be applied to a diverse type of data as long as the data are meaningful for a target application. In business, for example, DM can be used for segmenting customers into groups and use these different groups to target marketing and promotions. In healthcare, DM can be used for supporting physicians in identifying effective treatments and best practices, so that patients receive better and affordable healthcare services (Koh & Tan, 2011). In a similar way, DM has found a wide application in the education industry, particularly, in the improving of teaching and learning practices. In fact, DM has gained a huge success in education that has resulted in the creation of the International Educational Data Mining Society (IEDMS), with a mission to promote research in the emergent subfield of DM, the so-called Educational Data Mining (EDM), and also to organize the EDM conferences and journal.

Various definitions have been formulated to refer to the term ‘Educational Data Mining’ or EDM. According to the Journal of EDM (“International Educational Data Mining Society,” 2016), “*EDM is an emerging discipline, concerned with developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learn in*”.

Calders and Pechenizkiy (2012) has defined EDM as "*both a learning science, as well as a rich application area for data mining, due to the growing availability of educational data. It enables data-driven decision making for improving the current educational practice and learning materials*".

Other similar definition, but stressing the use of DM, was suggested by C. Romero and Ventura (2010), which defined EDM as the application of DM techniques to educational data, in order to resolve educational research issues.

Although these definitions may differ in some details, they all share an emphasis on the application of DM on the data exclusively coming from educational context, in order to address educational issues (Nabila Bousbia & Belamri, 2014). Essentially, as illustrated in Calders and Pechenizkiy (2012) (see Figure 2.1), EDM is an interdisciplinary discipline made up of a combination of the following areas: Computer science, Statistics, and Education. The intersection of these three areas also forms other subareas closely related to EDM such as computer-based education, DM and Machine Learning, and Learning Analytics (LA) (Cristobal Romero & Ventura, 2013).

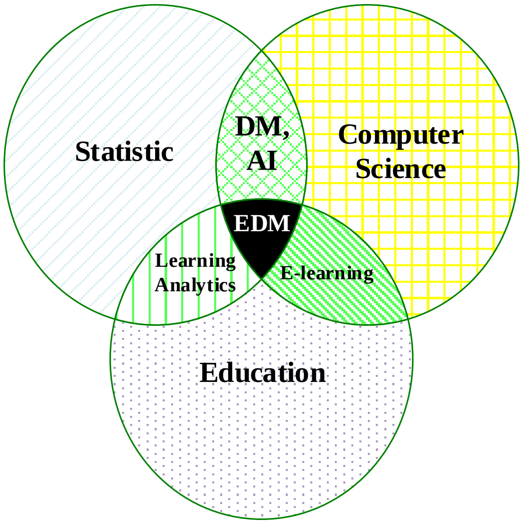


Figure 2.1: Areas in relation with EDM (Cristobal Romero & Ventura, 2013)

2.2.1. Objectives of the EDM

Taking the previous examples, DM in business field seeks to increase profit, whereas in healthcare, seeks to save more lives. These are tangible goals that can be measured in terms of numbers. In contrast, in education, DM seeks to improve the teaching-learning practices (C. Romero & Ventura, 2010), which is a more general goal, and sometimes can be difficult to quantify. However, if look at this goal from a different perspective, from the stakeholder’s (learners, educators, administrators, researchers) viewpoint, becomes clear that the objective of EDM is to improve users' experiences within the educational systems.

Following are presented some examples of the EDM goals categorized according to educational user viewpoint (Cristobal Romero & Ventura, 2013):

- Learners: To assist a learner’s reflections on the situation, to provide adaptive feedback or recommendations to learners, to respond to student’s needs, to improve learning performance, etc.
- Educators: To understand their students’ learning processes and reflect on their own teaching methods, to improve teaching performance, to understand social, cognitive and behavioural aspects, etc.
- Researchers: To develop and compare data mining techniques to be able to recommend the most useful one for each specific educational task or problem, to evaluate learning effectiveness when using different settings and methods, etc.
- Administrator: To evaluate the best way to organize institutional resources (human and material) and their educational offer.

2.2.2. The process of applying EDM

The process of applying EDM is very similar to the one of the KDD (Barahate, 2012; Cristobal Romero & Ventura, 2013; Sachin & Vijay, 2012). The process consists of three major tasks:

- Data pre-processing. It is the starting point of the EDM process and consists of collecting data from the educational environment and then preparing it for later use. Typically, data pre-processing tasks consume over 60% of the time, effort and resources employed in the whole EDM process (Cristóbal Romero et al., 2014).
- Data mining. Once the data pre-processed, the appropriate EDM method and technique is applied.
- Result interpretation, also known as post-processing. It is the interpretation and the assessment of the obtained results.

As shown in Figure 2.2, EDM can be seen as an interactive cycle of hypothesis formation, testing, and refinement.

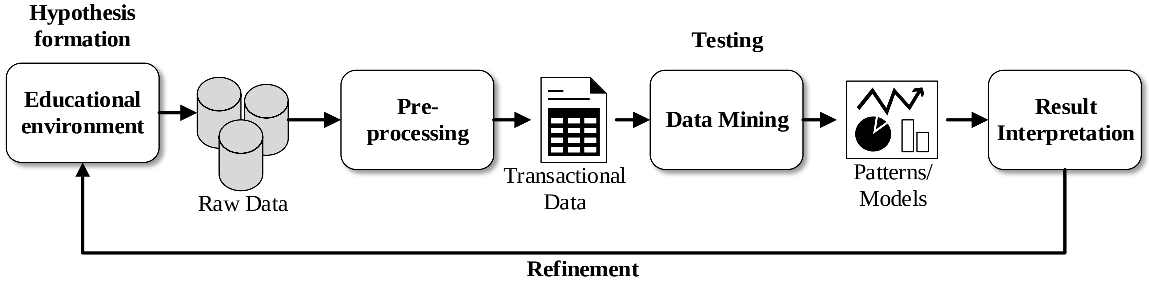


Figure 2.2: Educational knowledge discovery and data mining process; Adapted from (Cristobal Romero & Ventura, 2013)

2.2.3. EDM and Educational environments

DM can be applied to data coming from both types of educational environments, namely, traditional classroom and computer-based education. Traditional classroom environments refer to the type of educational system that is based mainly on face-to-face contact between educators and students organized through lecturers, class discussion, small groups, individual seat work, etc. (Cristobal Romero & Ventura, 2013). Some example of traditional classroom environments are primary schools, secondary, higher education, etc. These kinds of educational systems usually keep their records (student’s information, educator’s information, class and schedule information, etc.) on diverse traditional databases such as hard copy files, online web pages, multimedia databases.

On the other hand, computer-based education, also known as distance learning or simply E-learning, consists of techniques and methods providing access to educational programs for students who are separated by time and space from lecturers (C. Romero & Ventura, 2007). Some example of E-learning environments are Intelligent Tutoring System

(ITS), Learning Management System (LMS), Adaptative Educational Hypermedia System (AEHS), Computer Supported Collaborative Learning (CSCL), serious games, test and quiz systems, etc. Such systems typically store all the system’s information (user’s profile, user’s interaction data, etc.) in digital databases such as relational databases and log files.

Traditional classroom and E-learning environments have different characteristics, for example, each one of them provides different data sources that have to be pre-processed in different ways depending on both the nature of available data and the specific problems and tasks to be resolved by EDM techniques (C. Romero & Ventura, 2007).

Besides, traditional classroom systems can also use E-learning systems as a complementary tool to face-to-face sessions. In such cases, there are studies that combined the two data sources in order to give a complete view of the learner's behaviour and performance. As an example, Cristóbal Romero et al. (2010) attempted to predict the success of students in the final exam based on their participation level in online forums.

Although there are studies that apply DM on data gathered in traditional classroom environments and data coming from institutional administrative systems, most of the research papers presented in EDM conferences deal with DM techniques applied on data coming from Learning Management Systems (LMS), Intelligent Tutoring Systems (ITS) and AdaptivE Hypermedia Systems (AEHS) (Nabila Bousbia & Belamri, 2014; Kleftodimos & Evangelidis, 2013; C. Romero & Ventura, 2010). The following subsections describe the characteristics of these three types of E-learning systems.

2.2.3.1. Learning Management Systems

A learning Management System (LMS) is a form of E-learning based on a Web platform for delivery of education through technology, administration, documentation, tracking, reporting of training programs, classroom and online events, courses, and training content (Cristóbal Romero et al., 2014). LMSs also offer a great variety of channels and workspaces to facilitate information sharing and communication among all the participants in a course, let educators distribute information to students, produce content material, prepare assignments and tests, engage in discussions, manage distance classes and enable collaborative learning with forums, chats, file storage areas, news services, etc. (Cristobal Romero & Ventura, 2013). LMSs accumulate a vast amount of information which could create a gold mine of educational data. They can record any student activities involved such as reading, writing, taking tests, performing tasks in real, and commenting on events with peers (Cristobal Romero, Espejo, Zafra, Romero, & Ventura, 2010). Some examples of commercial LMSs are Blackboard and Virtual-U while some examples of free LMS are Moodle, Ilias, Sakai, and Caroline. Note that among the available computer-based learning environments, Moodle was the most tested learning environment in EDM researches (Nabila Bousbia & Belamri, 2014).

2.2.3.2. Intelligent Tutoring Systems

Intelligent Tutoring Systems (ITS) is another form of E-learning systems that aims to provide immediate and customized instruction or feedback to students by modelling student behaviour and changing its mode of interaction with each student based on its individual model. Normally, it consists of a domain model, student model, and pedagogical model. ITS record all student-tutor interaction such as mouse clicks, typing, and speech (Cristobal Romero & Ventura, 2013).

ITS has the ability to log and pool detailed, longitudinal interactions with large numbers of students which create a huge source for educational data sets (Mostow & Beck, 2006). "*Although ITSs record all student-tutor interaction in log files or databases, there are some other data stores available within an ITS, for example, a domain model that incorporates a set of constraints relevant to the tutor's domain, a pedagogical data set that* *contains a set of problems and their answers, and a student model that stores information about each student with respect to all the constraints, satisfactions, and violations recorded*" (Cristóbal Romero et al., 2014).

2.2.3.3. Adaptive Hypermedia Systems

Adaptive and Intelligent Hypermedia Systems (AIHS) are one of the first and most popular kinds of adaptive hypermedia and provide an alternative to the traditional just-put-it-on-the-Web approach in the development of educational courseware (C. Romero & Ventura, 2010). "*They attempt to be more adaptive by building a model of the goals, preferences, and knowledge of each individual student and using this model throughout the interaction with the student in order to adapt to the needs of that student. The data coming from these systems is semantically richer and can lead to a more diagnostic analysis than data from traditional Web-based education systems. In fact, the data available from AIHs are similar to ITS data; that is, AIHs store data about the domain model, student model and interaction log files (traditional Web log files or specific log files)*" (Cristóbal Romero et al., 2014).

The above educational environments provide a gold mine of educational data that can be turned into useful information through the application of DM methods and techniques. In EDM, these methods can be roughly divided into eleven categories. The next section will provide a detailed description of these methods and the most relevant studies.

2.2.4. EDM methods and techniques

Because EDM is a subfield of DM, EDM has inherited most of its methods and techniques from DM. Nevertheless, educational systems have special characteristics that require a different treatment of the mining problems (Cristobal Romero & Ventura, 2013). Although most of the traditional DM techniques can be applied directly, others cannot and have to be adapted to the specific educational problem (C. Romero & Ventura, 2010). Thus, the EDM taxonomy includes (Nabila Bousbia & Belamri, 2014; Cristobal Romero & Ventura, 2013):

- The traditional DM methods, namely Prediction methods, Clustering, Outlier Detection, Relationship Mining, Social Network Analysis, Process Mining, and Text Mining;
- And also, specific methods coming from EDM such as the Distillation of Data for Human Judgment, Discovery with Models, Knowledge Tracing, and Matrix Factorization.

2.2.4.1. Prediction

The aim of prediction-oriented methods is to construct a model which can infer a single aspect of the data (predicted variable) based on some combination of other aspects of the data (predictor variables) (Nabila Bousbia & Belamri, 2014). There are three techniques that are often used for predictive tasks: classification, regression, or density estimation. Classification is a technique that can be used when the predicted variable has a discrete value. Regression is a technique that can be used when the predicted variable has a continuous value. On the other hand, density estimation is a technique that can be used when the predicted value is a probability density function. Some examples of studies applying predictive methods in EDM are (Dole & Jayant, 2014; Jovanovic, Vukicevic, Milovanovic, & Minovic, 2012).

2.2.4.2. Clustering

The goal of clustering is to identify a set of categories or clusters embedded in the data. Each cluster represents a collection of data objects that, in some sense, are similar to one another (Cristobal Romero & Ventura, 2013). Typically, some kinds of distance measures are used to decide how similar instances are (Cristobal Romero & Ventura, 2013). Once a set of clusters has been determined, new instances can be classified by determining the closest cluster (Nabila Bousbia & Belamri, 2014). In EDM, clustering has been used by Jovanovic et al. (2012) for grouping students based on their cognitive styles in E-learning environment.

2.2.4.3. Relationship Mining

Relationship mining is used to reveals interesting relationships among variable in a given dataset. There are different types of relationship in mining techniques such as Association Rule Mining (also known as Link analysis), Sequential Pattern Mining, Correlation Mining and Causal Data Mining (Nabila Bousbia & Belamri, 2014). Association Rule Mining are used to find relationships among data and presents them in the form of rules. Sequential Pattern Mining are used to find temporal associations between variables. Correlation Mining is used to find (positive or negative) linear correlations between variables. And, Causal Data Mining, attempts to find whether one event (or observed construct) was the cause of another event (or observed construct), either by analysing the covariance of two events or by using information about how one of the events was triggered (Barahate, 2012). In EDM, Association rules and Sequential Pattern Mining has been used to enhance web-based learning environments in order to improve the degree to which the educator can evaluate the learning process (Zaïane, 2001). Also, relationship mining has been used to identify relationships between the students' online activities and the final marks (Cristóbal Romero et al., 2010).

2.2.4.4. Outlier Detection

The goal of outlier detection is to discover data points that do not comply with the general behaviour of the data (Cristóbal Romero et al., 2014). The general approach for finding these outliers is the use of distribution and deviation analysis (Altorf, 2007). In EDM, outlier detection has been used to detect deviations in the learner's or educator's actions or behaviours, irregular learning processes, and for detecting students with learning difficulties (Barahate, 2012).

2.2.4.5. Social Network Analysis

Social Network Analysis (SNA) or structural analysis, aims at studying relationships between entities in a networked information (Cristobal Romero & Ventura, 2013). The SNA techniques view social relationships in terms of network theory consisting of nodes, and connections/links, where, the nodes represent the individual actors within the network and the connections/links represent the relationships between the individuals which might be in the form of friendship, kinship, organizational position, etc. (Cristobal Romero & Ventura, 2013). In EDM, Social network analysis has been used as a method for analysing interaction in collaborative online learning environments (Doran, Doran, & Mazur, 2011). Rabbany, Takaffoli, and Za (2011) also proposed the use the data mining and social networks for interpreting and analyse the structure and relations in collaborative tasks and interactions with communication tools.

2.2.4.6. Process Mining

Process mining techniques attempt to extract knowledge from event logs recorded by an information system in order to have a clear visual representation of the whole process (Cristobal Romero & Ventura, 2013). The Process Mining consists of three subfields: conformance checking, model discovery, and model extension. In EDM, process mining has been used for reflecting students' behaviours in terms of their examination traces consisting of a sequence of course, grade, and timestamp triplets for each student (Trčka, Pechenizkiy, & Aalst, 2011).

2.2.4.7. Text Mining

Text mining is the process of analysing text to extract information that is useful for particular purposes. This technique is useful to find interesting patterns from semi-unstructured or unstructured datasets such as full-text documents, emails, HTML, chat messages, etc. (Sachin & Vijay, 2012). Text mining tasks include text categorization, text clustering, entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modelling (Cristobal Romero & Ventura, 2013). In EDM, text mining has been used to analyse the content of discussion boards (Ueno, 2004), forums (Dringus & Ellis, 2005), documents (Tane, Schmitz, & Stumme, 2004), online questions and chats (He, 2013).

2.2.4.8. Distillation of Data for Human Judgment

It is a technique of EDM concerned with making complex data understandable by humans to leverage their judgment. This approach uses summarization, visualization, and interactive interfaces to highlight useful information and support decision-making (Cristobal Romero & Ventura, 2013). Basic statistics such as mean and standard deviations are a simple example of the summarization (Cristobal Romero & Ventura, 2013). Spreadsheet charts, scatter plot, 3D representations, and other visual representations, are some examples of visual tools that can be used to depict the data in a way that enables a human to quickly identify or classify features of the data (Cristobal Romero & Ventura, 2013). Mazza and Dimitrova (2004) has proposed a visualization tool called CourseVis that can be embedded into WebCT to visualise graphically the data of an on-line distance course. Listen (Mostow et al., 2005) is another project that uses student-tutor interactions logs to generate graphical representations, which can be used by instructors to gain an understanding of their learners.

2.2.4.9. Discovery with Models

Discovery with models attempts to construct a model via prediction, clustering or knowledge engineering, then use this model as a component in further analysis such as prediction or relationship mining(Cristobal Romero & Ventura, 2013). These methods have been used for identifying the relationship between student's behaviour and characteristics or contextual variables (Bouchet, Azevedo, Kinnebrew, & Biswas, 2012). Besides that, Discovery with Models can be useful to find, for example, how variations in intelligent tutor design impact students' behaviour over time (Jeong & Biswas, 2008), or how different types of student behaviours impacts students' learning in different ways (Cocea, Hershkovitz, & Baker, 2009).

2.2.4.10. Knowledge Tracing (KT)

In EDM, Knowledge Tracing (KT) was proposed as a method for estimating student's mastery skills in a cognitive tutor system on the basis of cognitive model and logs of students' responses (Corbett & Anderson, 1995). The cognitive model is used to map a problem-solving item to the skills required, whereas the logs are used to store the correct and incorrect answers of a student. Together, they are used as evidence of student knowledge on a particular skill. KT tracks student knowledge over time and it is parameterized by variables. There is an equivalent formulation of KT as a Bayesian network (Cristobal Romero & Ventura, 2013)

2.2.4.11. Matrix Factorization

Matrix Factorization is a factorization of a matrix into a product of matrices. In other words, Matrix factorization is the task of approximating a matrix X by the product of two smaller matrices W and H, i.e $X \approx WH^T$ (Koren, Bell, & Volinsky, 2009). Non-negative Matrix Factorization (NMF) is a form of Matrix Factorization that consists of a matrix of positive numbers, as the product of two smaller matrices. In the context of education, Matrix Factorization has been used as a mechanism for producing recommendations as well as to support in student's performance prediction (Thai-Nghe et. al, 2010). On the other hand, NMF has been used as a mechanism for assessing student's cognitive skills in an intelligent learning environments (Desmarais, 2012). The proposed NMF approach relies on a mapping of tasks to skills. It uses a matrix M that represents the observed examinee's test outcome data that can be decomposed into two matrices: Q that represents the *Q-matrix* (question matrix) of items and S that represents each student's mastery of skills.

One of the most commonly used method in EDM is prediction. In particular, student's performance prediction methods, because they enable educators and learners to forecast the future, thus, allowing them to conduct in-time interventions towards to prevent underachievement. The next section reviews the current state of EDM research in student' performance prediction methods.

2.2.5. EDM methods for predicting student performance

The goal of student’s performance prediction methods is to estimate the unknown value (e.g. student performance, knowledge, score, mark) of a variable that describes the student (C. Romero & Ventura, 2010). In EDM, these methods are very important for conducting in-time interventions for the students, whose performances are not at the expected level, especially the students who are more likely to fail academically, and may drop out (Moradi, Moradi, & Kashani, 2014). Student’s performance prediction methods can be used, for instance, by both educators and learners to predict the student's final outcome in a course. This allows them to take proactive actions towards to improve their teaching-learning practice, also, allows adaptation of the educational materials and strategies throughout the learning process.

So far, several methods for predicting student performance were proposed in EDM, where, a majority of them were applied in LMS and ITS (Cristóbal Romero et al., 2014). Cristobal Romero et al. (2010) have demonstrated how the web usage mining can be applied in Moodle E-learning systems to predict the marks that university students will obtain in the final exam of a course. They conducted an experiment in order to obtain, on the one hand, an accurate model for predicting student's final marks based on the students' usage data in Moodle LMS. On the other hand, a model that is comprehensible for non-expert users such as an instructor. The result of their findings suggested that decision trees, rule induction, and fuzzy rule were the most suitable algorithms for educational context because they provide comprehensible models that are easy to interpret.

Similarly, Cristóbal Romero et al. (2010) explored the use of Rare Association Rule Mining (RARM) over educational data gathered from the Moodle system for discovering unusual student behaviours that can influence in the student performance. They studied students' usage data to evaluate the relationship between the on-line activities and the final mark obtained by the students. They discovered a set of rules that enabled them to predict students’ exam results (fail or pass) on the basis of student interaction with Moodle activities such as quizzes, assignments and forum activities.

In (Thai-Nghe et al., 2010), a recommender system was proposed for predicting student performance in computer-aided tutoring systems. They used students’ interactions log files to build success and progress indicators and afterwards a matrix factorization technique were applied for predicting student performance.

Lauría and Baron (2011) have used demographic data and logging data collected from the Sakai E-learning system to develop predictive models of students' success. They used different techniques such as C4.5/C5.0 decision trees, Support Vector Machine (SVM), Bayesian network, factor analysis and logistic regression, for building mining models to predict student performance.

Jovanovic et al. (2012) have proposed a tool that uses classification models for prediction of students’ performance, and cluster models for grouping students based on their cognitive styles in Moodle. The study claims that clustering students based on cognitive styles and their overall performance will enable better adaptation of the learning materials with respect to their learning styles. Thus, they developed a Moodle plugin for predicting student’s performance that can be embedded into Moodle LMS. Educators can use this tool to browse through a list of students involved in their course and see the prediction of success for each student on that course. These predictions helped the stakeholders such as teachers, students and business people, for early engaging with students who are likely to become excellent on a selected topic. Also, enabled educators to adapt their teaching approach towards to the students that are predicted to be unskilled.

López, Luna, Romero, and Ventura (2012) proposed a classification via clustering approach to predict the final marks in a university course on the basis of Moodle forum data. This approach used classification and clustering to predict students' final marks from their participation in forums.

The authors in (Moradi et al., 2014) designed a multi-channel decision fusion approach to estimate the overall student performance. The proposed approach attempts to estimate the overall student performance based on the performance achieved in different assignments, such as homework assignments, coding projects, hardware projects, quizzes, laboratory, essay, exams, and team projects. The main advantage of the proposed approach is in its ability to predict the students' performance after a few assignments. Therefore, helping the instructors better manage their class and adjust educational materials to prevent underachievement.

In summary, different DM techniques and models were used for building students' performance prediction methods. Regarding student's performance studies, a special emphasis was given to E-learning environments, since it tries to compensate the shortcomings caused by the lack of face-to-face interaction with a human expert (Moradi et al., 2014). Furthermore, computer-based environments provide various sets of data that can be useful for predicting students' performance, for example, student's usage data (i.e. quizzes, assignments and forum activities), web usage data, demographic data, forum participation data, logging data, etc.

Note that, during the process of building predictive models, several heterogeneous tools are used for, on the one hand, collecting the data from the educational environments. On the other hand, for applying the EDM methods and visualizing the obtained results. The next section describes some of the most commonly used tools among the EDM researches.

2.2.6. EDM tools

As previously mentioned in section 2.2.1, EDM is a three steps process that consist of the following tasks: data pre-processing, data mining and post-processing. Several tools have been proposed for the execution of each of these tasks. In general, these tools can be grouped into two categories: general purpose tools and specialized tools for EDM.

2.2.6.1. General purpose tools

General purpose tools usually have a wider scope of domains. In other words, they can be used to perform DM in a large number of domains, including educational contexts. These tools can support almost every aspect of the KDD process such as data pre-processing, data mining and post-processing. General purpose tools can be roughly grouped into the following categories (Nabila Bousbia & Belamri, 2014):

- Data analysis tools. These tools are statistical software packages that also can be used for pre-processing data (Cristóbal Romero et al., 2014). The most commonly used data analysis tools in EDM are Matlab, R (including Rattle and RDatamining), IBM SPSS Modeler (formerly, Clementine), SAS (Enterprise Miner), Statistica and Microsoft Excel.
- Data mining tools. These are specialized tools for data mining. Usually they provide features such as data preparation or data transformation, feature selection, and meta-data handling. In EDM, open source data mining software were commonly used because they offer powerful tools for users in education and researchers (X. Chen, Williams, & Xu, 2007). There is a large variety of open source software solutions in this field, being Weka (Hall et al., 2009), RapidMiner (“RapidMiner,” 2015) and Knime (“Knime,” 2015).

- Business intelligence software applications. These tools are focused on business data analysis and visualization for supporting decision making, and oriented to the treatment of large amounts of data (Cristóbal Romero et al., 2014). They provide advanced functionalities for data exploration and transformation. Some examples of Business Intelligence software are Orange ("Orange," 2015), and Angoss ("Angoss," 2015).

General purpose tools also come with some disadvantages. First, they all run on a single machine or centralized machine. Second, since they are for the general purpose, sometimes they can be inadequate for solving specific educational problems. Mainly because of the following facts (C. Romero & Ventura, 2010): the application of DM in different area have different goals; the data in educational environments is very different from the data of others domains; and there are specifics EDM techniques, although most of the traditional DM techniques can be applied directly, others cannot and have to be adapted to the specific educational problem.

Finally, it is difficult and tedious to prepare the data to be used in these tools. Typically, the process of preparing the data to be used in these tools takes time, requires huge efforts and consume lots of resources. That is why; many researchers in EDM have demanded the creation of specialized EDM tools, in order to simplify the development of EDM researches and practices.

2.2.6.2. Specialized tools for EDM

In response to the increasingly need for specialized EDM tools, many tools were proposed in EDM community. In general, the currently existing tools were designed especially for computer-based environments, because, physically, these environments provide supportive tools (digital data, programmable environments, etc.) that enable the automation of some EDM tasks. In Table 2-1, is presented a summary of some of the most recent tools.

Table 2-1: Some tools for EDM application

Tool and references	Goal
MMT (C Romero & Ventura, 2010)	To facilitate the execution of all data mining steps in Moodle data
ADE (Jovanovic et al., 2012)	To allows automatic extraction of Moodle data.
AAT (Graf, Ives, Rahman, & Ferri, 2011)	To analyse student’s behaviour in learning systems
E-learning Web Miner (Zorrilla & García-Saiz, 2012)	To discover student’s behaviour profile in virtual course
SNAPP (Bakharia & Dawson, 2011)	To display the relation evolution between participants in discussion forums
EDM visualization tool (Johnson & Barnes, 2010)	To visualize the process in which students solve procedural problems in logic
DRAL (Zafra, Romero, & Ventura, 2013)	To discover relevant e-activities for learners
IDLS (Nabalia Bousbia, Rebaï, Labat, & Balla, 2010)	To analyse learner’s browsing behaviour and learning styles
eLAT (Dyckhoff, Zielke, Bültmann, Chatti, & Schroeder, 2012)	To explore and correlate learning object usage, user properties, user behaviours, as well as assessment results based on graphical indicators

Although an effort is being made to provide tools for mining educational data, currently existing tools are still too complex for educators, EDM researchers, and users who are not expert in data mining (C. Romero & Ventura, 2010). In addition, the existing

EDM tools are just prototypes, most of which provide restricted features, tend to fulfil one research goal, and above all, they do not meet all EDM community requirements (Nabila Bousbia & Belamri, 2014; Cristóbal Romero et al., 2014; Cristobal Romero & Ventura, 2013). In response to these limitations, this study attempts to build an EDM tool that comforts with the EDM community requirements. With this goal in mind, this research conducted an in-depth review of literature aimed at capturing the requirements that meet with EDM community needs, which has resulted in the following:

- Reusability. The ability to be generic and re-usable tools that can be applied to any E-learning system and any data mining framework. Thus, a standardization of input data and output model would easily take place, as well as pre-processing, knowledge discovery and post-processing tasks (F. M. D. A. Ali & Ng, 2015; Nabila Bousbia & Belamri, 2014; C. Romero & Ventura, 2010; Cristobal Romero & Ventura, 2013).
- Extensibility. The ability to handle growth efficiently. EDM is not a static discipline, new algorithms are being created, as well as new datasets and models. Therefore, an effort should be made to make EDM tools flexible to new changes (C. Romero & Ventura, 2010; Cristobal Romero & Ventura, 2013).
- Usability. EDM tools need to be easy to use and learn. They need to provide a different degree of complexity accordingly user’s expertise or objectives. So that, both non-expert and expert users will not face difficulties while using this tools. Also, all EDM tasks (pre-processing, data mining, and post-processing) must be carried out in a single application with a similar interface (Nabila Bousbia & Belamri, 2014; C. Romero & Ventura, 2010; Cristóbal Romero et al., 2014; Cristobal Romero & Ventura, 2013).
- Maintenance and integration. EDM tools need to be easy to maintain. EDM tools are complex systems that typically integrates many other tools such as LMS, DM tools, and visualization tools. This complexity makes maintenance harder. Therefore, it is important to take into account the maintenance issue while designing

such tools. One possible solution to tackle this issue is the adoption of modular design (F. M. D. A. Ali & Ng, 2015; DeFreitas & Bernard, 2014).

Among the above requirements, the usability, reusability, and extensibility requirements were the most requested in EDM community. Usability, because of the nature of users found in educational contexts usually have few or almost no expertise in data mining analysis. So, making these tools available to the different educational actors (teachers, designers, administrators, and researchers) has become a ground requirement for the actual usage of these tools. Reusability requirement, on the other hand, was requested for making EDM tools more generic and able to deal with multiple environments, data, and models. One example of the fulfilment of reusability requirement would be the decoupling of EDM tools with a particular LMS or DM tool. This would make possible the reuse of this tool in fulfilment of different research goals. At last, the EDM community demanded tools more flexible to changes. In other words, tools that can easily accommodate changes or new developments being made in the community.

However, to our best knowledge, none of the existing tools meet all the requirements stated above. Therefore, this study introduces the distributed knowledge discovery technology to attempt to solve the limitations found in current EDM tools. The use of distributed knowledge discovery technology appears as a solution to the pointed limitations because it provides important features such as extensibility, aggregation, scalability, resilience and the ability to hide implementation details, and hence provide the basis for a well-architected distributed knowledge discovery applications.

The ultimate goal of this study is to build a distributed EDM system where the E-learning and data mining functionalities can be integrated into a third party application, through which can be hidden the complexity of technical details; therefore, simplifying the development of EDM in E-learning environment.

On account of this objective, this study conducted a survey of literature aimed at finding better solutions for implementing a distributed knowledge discovery for an EDM system. The next section provides a detailed description the modern distributed knowledge discovery solutions and the most relevant studies.

2.3. Distributed knowledge discovery

Distributed knowledge discovery, also known as Distributing Data Mining or simply DDM, exploits distributed architectures with functionalities for identifying, accessing, and orchestrating local and remote data, resources and mining tools into task-specific distributed workflows (Talia & Trunfio, 2012). The idea of DDM resources within a computer network is not new. The rapid evolution of technology and internet has demanded such modern DM solution. Past efforts attempting to embody this concept have resulted in a number of different distributed data mining implementations. There are mainly four types of Distributed Data Mining, namely DDM based parallel data mining agent (PADMA), DDM based on Meta-learning, DDM based on Grid, and DDM based on SOA. In the next subsections, follows the description of these types/categories of DDM and the most relevant studies.

2.3.1. DDM based Parallel Data Mining Agent (PADMA)

PADMA is a multi-agent based architecture for DM. Typically this type of system is composed of multiple intelligent DM agents, which assists users in performing complex tasks, such as accessing, analysing, and discovering the hidden patterns within the data warehouse (Liu, Cao, & He, 2011). The agents cooperate with each other and share same repository or meta-data. The related most significant systems include CAKE system developed by Khan (2008), EMADS (Albashiri & Coenen, 2009), and MAKMS (Z. Chen, Liui, & Liu, 2008).

Classifying, Associating and Knowledge discovEry, or simply CAKE, is a 4-tier architecture where the DDM is implemented using PADMAs and centralized metadata. The metadata contains all the rules of Classification and Association along with its data structure details. CAKE has a web interface used to provide the users with the interface to view the result (Khan, 2008).

Extendible Multi-Agent Data mining System (EMADS), is a multi-agent system where intelligent DM agents cooperate with one another to address DM requests under a decentralized control. The system currently comprises a set of agents, each delivers different functionality, including data agents along with user agents, task agents, mining agents, and housekeeping agents. The housekeeping agents provide facilities to execute the operation of EMADS, more specifically, it holds an Agent Management System (AMS) agent which controls the life cycles of other agents, and a Directory Facilitator (DF) agent which provides an agent lookup service. The current functionality of EMADS is limited to classification and Meta association rule mining (Albashiri & Coenen, 2009).

The Multi-Agent Knowledge Management System Model for Pervasive Computing (MAKMS), is a multi-agent system proposed for solving some major problems faced by traditional centralized DM models such as network limit, data privacy, and system incompatibility. The system has an intelligent DM agent (DMA) which is the core of the system, and is responsible for executes the mining tasks. Additionally, the system has an agent coordinator which is responsible for coordination and cooperation among mining agents as well as negotiates and synchronizes among the modules (Z. Chen et al., 2008).

2.3.2. DDM based on Meta-learning

The meta-learning technique aims at building a global model from a set of inherently distributed data sources (Talia & Trunfio, 2012). Meta-learning is loosely defined as learning from learned knowledge (Sen, Dash, & Pattanayak, 2012). In classification scenario, meta-learning, is achieved by learning from the predictions of a set of base classifiers on a common validation set (Talia & Trunfio, 2012).

Figure 2.3 illustrates the different steps to build a global classifier from a set of distributed training sets using a meta-learning approach. According to Talia and Trunfio (2012), Meta-learning is basically a three-step process: first, a number of base classifiers are generated by applying learning programs to a collection of distributed datasets in parallel. Next, a meta-level training set is formed by combining the predictions of the base classifiers on a common validation set. Finally, a global classifier is trained from the meta-level training set by a meta-learning algorithm.

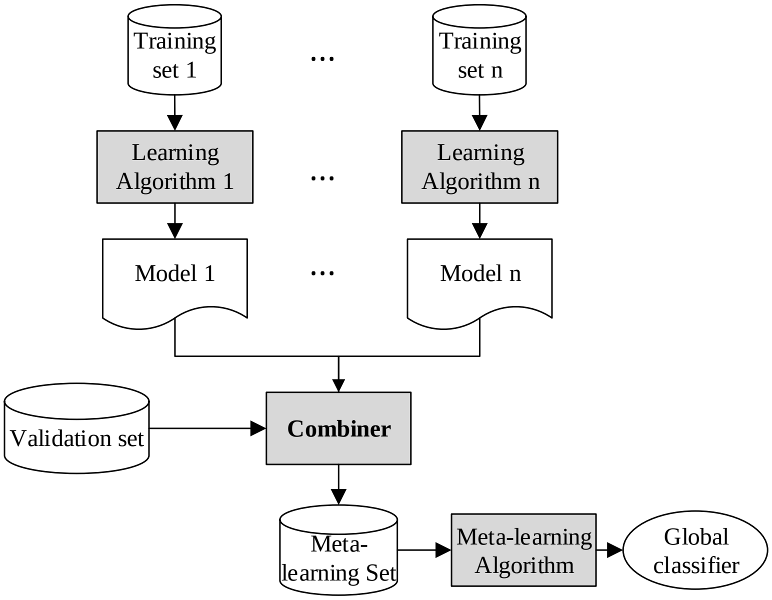


Figure 2.3: Building a global classifier with meta- learning (Talia & Trunfio, 2012)

So far, JAM (Java Agents for Meta-Learning) is the most representative system employing meta-learning as a means for mining databases (Sen et al., 2012). JAM is a distributed agent-based data mining system that provides a set of meta-learning agents for combining multiple models that were learned at different sites (Zeng et al., 2012). JAM is flexible, scalable, portable and extensible data mining facility (Stolfo et al., 1997).

EEMS (Execution Engine of Meta-learning System for KDD in Multi-agent Environment) is a system proposed by P. Luo, He, Huang, and Lin (2005) that also adopted meta-learning. The system considered an execution engine as the kernel of the system to provide mining strategies and services. The engine has an extensible architecture based on a mature multi-agent environment. It connects different computing hosts to support intensive computing and complex process in a distributed fashion. Reuse of existing mining algorithms is achieved by encapsulating them into agents. The algorithms also define a DM workflow as the input of the engine and detail the coordination process of various agents to process it.

2.3.3. DDM based on Grid

DDM on grid aims at sharing resource via local and wide area networks (Liu et al., 2011). The Grid computing paradigm is receiving an increasing attention from several industries, businesses and research communities due to its capability to provide resource access and computing power delivery to large organizations. The term Grid comes from an analogy to the “electric power grid” (Talia & Trunfio, 2012). As the electric power grid delivers a universal and standardized access to electricity to individuals and industries, the goal of computational Grids is to enables the sharing, selection, and aggregation of a wide variety of geographically distributed resources including supercomputers, storage systems, data sources, and specialized devices owned by different organizations for solving large-scale resource intensive problems in science, engineering, commerce, etc. (Braz, 2005).

The need for integration and interoperability among a different number of applications has led the Grid community to the design of the Open Grid Services Architecture (OGSA) as an extension of Web services (Talia & Trunfio, 2012). In fact, OGSA is an implementation of the Service Oriented Architecture model (SOA) (see section 2.5.4) within the Grid context (Cesario & Talia, 2008). In OGSA, every resource such as computers, storages, and programs are represented as a grid service that conforms to a set of conventions and supports standard interfaces for the development of interoperable Grid systems and applications (Talia & Trunfio, 2007). Likewise, Grid offers support for implementation and use of distributed knowledge discovery systems.

According to Brezany and Hofer (2003) a distributed data mining on Grid must comfort with the following requirements:

- Open architecture. In order to make the application more extensible for the grid community, the application must be based upon an open architecture such as OGSA.
- Data distribution, complexity, heterogeneity, and large data size. The application must be able to cope with highly dimensional data sets, large data sizes and geographically distributed and dispersed repositories.
- Compatibility with existing Grid infrastructure. The higher levels of the application must use basic grid techniques to be compatible with other grids.
- Openness to tools and algorithms. The application must be open to the integration of new data mining methods and techniques.

- Scalability. The application scalability in terms of a number of nodes used for performing the distributed knowledge discovery tasks, and in terms of performance achieved by using parallel computers to speed up the data mining task.
- Grid, network, and location transparency. End users should be able to perform the knowledge discovery task on Grid with less effort, in an easy and transparent way, without needing to know details about the underlying grid technique, the network features and the physical location of data sources.
- Security and data privacy. A key requirement of data mining is access to pertinent data (A. B. M. S. Ali & Wasimi, 2007) because commonly data mining tasks deal with very sensitive data. Therefore, the application must able to cope with user authentication, security, and privacy of data.
- Support OLAP (On-Line Analytical Processing) and data warehousing. OLAP is a technique of analysing data and looks for insights, whereas data warehousing refers to the integration of diverse databases. Data mining and OLAP together can be used in decision-making problems (Altorf, 2007). Typically, OLAP operations are done on the data warehouse instead of the normal database.

Several DDM systems based on Grid infrastructure have been designed and implemented. Following is discussed some of the most significant ones.

Stankovski et al. (2008) developed a DataMiningGrid environment for generic application and execution of knowledge discovery tasks. DataMiningGrid has been designed to meet the following requirements: handling of massive and distributed data, distributed operations, data privacy and security, user friendliness and resource identification and metadata. Also, the system includes additional features such as flexibility, extensibility, scalability, efficiency, conceptual simplicity and ease of use.

The GridMiner application (Brezany, Janciak, & Tjoa, 2005) is a knowledge discovery system based on Globus, using OGSA-DAI for access to Grid Data Sources. It is easy to use tool that integrates all aspects of knowledge discovery such as data cleaning, data integration, data transformation, data mining, pattern evaluation, knowledge presentation and visualization. GridMiner embraces an OGSA architecture in which a set of services are defined for knowledge discovery tasks and also providing OLAP support. According to A. S. Ali, Rana, and Taylor (2005) the number of analysis algorithms supported by GridMiner are very limited.

Cannataro and Talia (2003) proposed a software architecture for geographically distributed parallel and distributed knowledge discovery called Knowledge Grid. Knowledge Grid is based on OSGA model, using the Globus Toolkit 2. Knowledge Grid offers a high-level abstraction in which encompasses a set of service to support knowledge discovery tasks and providing resources. In Knowledge Grid, users can plan, store and re-execute their workflows as well as manage their results.

Other OGSA-compliant systems for distributed knowledge discovery on the Grid are WekaG (Perez, Sanchez, Herrero, Robles, & Pena, 2005) and Weka4WS (Talia, Trunfio, & Verta, 2008). Both exploit the well-known Weka toolkit and extended it to support distributed data mining on Grid. The first prototype of WekaG has only implemented Apriori algorithm, and as future work intends to extend WekaG by all the data mining algorithms included in Weka. The WekaG application implements the client-server architecture and some of its main features include coupling data sources, authorization access to resources, discovery based on metadata, planning and scheduling tasks and identifying the available and appropriate resources.

On the other hand, Weka4WS adopts the emerging Web Services Resource Framework (WSRF) for accessing remote data mining algorithms and managing distributed computations. Weka4WS encompasses all data mining algorithms for classification, clustering, and association provided by the Weka toolkit. Weka4WS extended the Weka's GUI (Graphical User Interface) by adding a method to do remote data mining tasks. Each task in the GUI is managed by a single thread; therefore, a user can start multiple tasks in parallel, therefore taking full advantage of the grid environment.

2.3.4. DDM based on SOA

A service-oriented architecture (SOA) is a style of building reliable distributed systems that deliver functionality as services, with the additional emphasis on the loose coupling between interacting services (Srinivasan & Treadwell, 2005). Services can be defined as a software component that can be accessed via a network to provide functionality to a service requester, and typically have the following characteristics (Ontario, 2011):

- Reusable. The ability to create services that can be used by multiple applications and other services.
- Autonomous units of business functionality. Each service delivers a business function that is independent of the other service.
- Contract-based. Interface and policies are strictly described by an interface specification.
- Loosely coupled. The ability to model services independently of their implementation. Therefore, allowing flexibility, scalability, replaceability and fault tolerance (Srinivasan & Treadwell, 2005). Flexibility refers to the ability to publish services on any server. Scalability refers to the ability of add or remove services according to demand. Replaceability refers to the capability of preserving the original interface even after a new or updated implementation is introduced. Finally, fault tolerance is a mechanism used when a service becomes unavailable for some reason. It allows clients to query the registry for alternate services that offer the required functionality, and continue to operate without interruption.

- Platform-independent. The service consumers and SOA services can interoperate even if implemented on different platforms. There must be an intermediary that handles the differences in data types between the SOA service and the requester.
- Discoverable and location independent. services are located through a service registry/catalogue and accessed via universal resource locators, and therefore may move over time without disruption to consuming systems
- Standards-based. Services are built, consumed, and described using standards such as Web Services Definition Language (WSDL), which provides information about the service, and Simple Object Access Protocol (SOAP), a packaging mechanism.

SOA is based on the interaction of the following actors (Talia & Trunfio, 2012):

- Provider, which is the server that provides the service;
- Registry, also known as a broker, which facilitate the connection between services and clients by allowing clients to locate the services of their interest;
- Clients or requestors, which requests and uses the services.

Figure 2.4 illustrates a simple service interaction cycle (Srinivasan & Treadwell, 2005), which begins with a service publishing itself through a well-known registry service (1). A potential client, which may or may not be another service, queries the registry (2) to search for a service that meets its needs. The registry returns a (possibly empty) list of suitable services, and the client selects one and passes a request message to it, using any mutually recognized protocol (3). In this example, the service responds (4) either with the result of the requested operation or with a fault message.

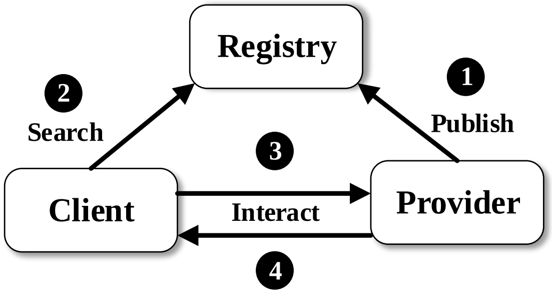


Figure 2.4: Actors and interactions in a service- oriented architecture (SOA) (Talia & Trunfio, 2012)

2.3.4.1. Web Services

A web service is an example of an SOA with a well-defined set of implementation choices. Web services are self-describing and modular business applications that expose the business logic as services over the Internet through programmable interfaces and using internet protocols for the purpose of providing ways to find, subscribe, and invoke those services (Nagappan, Skoczylas, & P. Sriganesh, 2003). Web services use the following standards for communication and data processing:

- XML (eXtensible Markup Language): an open standard data types and structure, independent of any programming language, development environment, or software system (Newcomer & Lomow, 2005).
- UUDI (Universal Description, Discovery, Integration specification): an XML-based standard for describing, publishing, and finding Web services.
- SOAP (Simple Object Access Protocol): an interoperability standard (Newcomer & Lomow, 2005), it uses XML-based standard for specifying how to exchange information between services and applications (Talia & Trunfio, 2012).

- WSDL: an XML-based standard used to describe the attributes, interfaces and other properties of a Web service. A WSDL document can be read by a potential client to learn about the service.

2.3.4.2. SOAP and REST Web services

Web services can be divided roughly into two groups, SOAP-based and REST-style. SOAP appeared as an attempt to overtake the older SOA technologies that were incompatible with the Internet, it was developed by Microsoft as an interoperability standard that exclusively relies on XML for its message format, using application layer protocols such as HTTP (HyperText Transfer Protocol) or SMTP (SOAP over Simple Mail Transfer Protocol) for message negotiation and transmission.

On the other hand, Roy Fielding proposed an alternative Web service style in his doctoral dissertation, which coined the acronym REST. REST stands for REpresentational State Transfer, it is a style of software architecture for distributed hypermedia system. Unlike SOAP, REST does not rely on XML to make a request or provide the response. Instead, REST uses URLs on its requests and can deliver the responses in different data representations such as Command Separated Value (CSV), JavaScript Object Notation (JSON) and Really Simple Syndication (RSS). Typically, REST systems (also known as RESTful) communicate over HTTP using the well-known HTTP methods for creating, reading, updating and deleting resources: POST, GET, PUT, and DELETE respectively.

SOA and Web services have been used to address the common problem of building DDM systems. Here are presented some examples of distributed knowledge discovery systems using service-oriented architecture and Web service.

An Extensible Service-Oriented Distributed Data Mining Framework (ESODDMF) was proposed by Kumar, et al. (2004), which is a DDM infrastructure that allows the integration of distributed, heterogeneous environment and complex interconnectivity. It is a four-tier architecture where the DDM is implemented using Web services with the purpose of hiding the complexity of implementation details and enabling users to perform data mining in a utility-like fashion. The first prototype was implemented with only three algorithms: Normalization, Fractal Dimension, and Apriori algorithm. ESODDMF provide a web interface which users can interact with it to perform mining activities.

Guedes, Meira, and Ferreira (2006) developed an Anteater platform which is a data mining service for decision support. It is based on a four-tier architecture that contains a set of servers offering their services through well-defined interfaces, including the: data server, mining server, application server, and visualization server. Anteater relies on Web services and parallelism to address interoperability, extensibility and scalability problems. Anteater uses visual metaphors to present the system's functionality to domain-oriented end users by keeping technical details transparent to them. To exploit parallelism while maintaining performance, Anteater provides a runtime system called Anthill. Anteater implements three different data mining technique: association rules with Apriori, classification with the ID3 algorithm, and clustering with K-Means algorithm. In addition, Anteater provides users with a simple interface that abstracts the algorithms' technical details.

Cheung et al. (2006) proposed a Service-Oriented Distributed Data Mining platform that exploits the Business Processing Language for Web services (BPEL4WS) and uses the learning-from-abstraction methodology to achieve privacy-preserving DDM. The system adopts WSDL standards to import/export functionalities over the internet, and BPEL4WS to specify the DDM processes and describe services flow. BPEL4WS provides a model and a grammar for specifying interactions between a business process and its partners through Web services interfaces. The learning-from-abstraction and self-adaptive approach to distributed data mining were demonstrated using different data mining applications, including clustering and manifold unfolding for visualization.

The Service Oriented Miner, or simply SOMiner (Birant, 2011) is another proposed system that relies on SOA and Web services to achieve DDM. SOMiner offers the necessary support for performing knowledge discovery tasks such as data pre-processing, data mining, post-processing, and visualization. Also, proposes Semantic Web Services for building a comprehensive high-level framework for distributed knowledge discovery in SOA models. SOMiner architecture contains six layers: data layer (DL), the application layer (AL), user layer (UL), data mining layer (DMSL), semantic layer, and complementary service layer (CSL). Users can interact with a graphical user interface provided by UL to publish services, design and submit data mining application, and visualize results.

Differently from the systems discussed above, the DRHPDM (Data source Relevance-based Hierarchical Parallel Distributed data mining Model) proposed by Liu et al. (2011), adopted both Web services and Multi-agent technologies to build a DDM for E-business. The combination of these technologies improved the openness, enabled cross-platform ability and enhance the intelligence of the DDM system.

2.3.5. Important remarks

Table 2-2 summarizes the surveyed literature on the topic of distributed knowledge discovery according to the type of DDM implementation. In general, most of the existing distributed knowledge discovery systems were proposed for general purpose use. In other words, they were developed to accommodate a wide range of existing application domains, for instance, e-business, e-science, e-commerce, etc. However, the claim that these DDM tools can accommodate a large scope of domains becomes partially true if considering that different domains have different characteristics, such as different computing, communication, data, storage sources, and so on. Therefore, from this research point of view, it is also important to take special consideration of the domain specification when designing the corresponding DDM system. In the case of education domain, for instance, DM problems have some special characteristics that require the issue of DDM to be treated in a different way (C. Romero & Ventura, 2010).

On the other side, a majority of research tackled the DDM issue by using Grid technology, because Grid solutions have allowed focus on large-scale resource sharing, innovative applications, and, in some cases, high-performance orientation (Zeng et al., 2012). Also, because Grid extends service-oriented approaches to its implementation, which allows the representation of resources within the Grid infrastructure as Web services. Conversely, Grid has the limitation of providing a rigid structure (Birant, 2011), closed-architecture (Liu et al., 2011), and Grid computing itself is relatively new, its standards and technologies are still evolving (Liu et al., 2011). On top of that, to the best of our knowledge, none of the existing E-learning systems makes uses of grid-based techniques, thus, it is likely to arise integration issues when adopting Grid for building a DDM for E-learning settings.

SOA and Web services, on the other hand, are more open and offers a generic approach that allows the integration of diverse systems (Birant, 2011). Rather than a rigid structure and closed architecture (Birant, 2011; Liu et al., 2011), SOA and Web services provide important features such as extensibility, aggregation, scalability, resilience and the ability to hide implementation details behind well-defined interfaces. Hence provide the basis for a well-architected distributed knowledge discovery applications (Srinivasan & Treadwell, 2005).

Table 2-2: Some DDM frameworks

Type of DDM	System and Reference
Parallel Data Mining Agent	CAKE (Khan, 2008), MAKMS (Z. Chen et al., 2008), EMADS (Albashiri & Coenen, 2009)
Meta-learning	JAM (Stolfo et al., 1997), EEMS (P. Luo et al., 2005), XCS (Dam, Abbass, & Lokan, 2005)
Grid	K-Grid (Cannataro & Talia, 2003), GridMiner (Brezany et al., 2005), FAEHIM (A. S. Ali et al., 2005), DMGA/WekaG (Perez et al., 2005), AGrIP (J. Luo, Wang, Hu, & Shi, 2007), ChinaGrid (Song, Wang, Xiong, & Jin, 2007), DataMiningGrid (Stankovski et al., 2008), Weka4WS (Talia et al., 2008)
SOA	ESODDMF (Kumar et al., 2004), Anteater (Guedes et al., 2006), SODDM (Cheung et al., 2006), SOMiner (Birant, 2011), ElWM (Zorrilla & García-Saiz, 2012)
Hybrid	DRHPDM(Liu et al., 2011)

2.4. Summary

EDM is a new growing research field that concerns with an exploration of educational data, and development of methods to apply in existing education systems in order to address educational issues. In computer-based education environments, EDM has found a wide use, particularly in the enhancement of the following aspects: (1) provide learners/students the opportunity to predict their performance; (2) allow educators to better assist their learners; (3) assist educational administrators in designing better strategies to reduce the costs of educative personalization and adaptation. EDM is a three steps process: (1) data pre-processing, (2) data mining, and (3) post-processing tasks.

Prediction methods are commonly used in EDM researches. Particularly, student’s performance prediction methods, since it allows educators and learners to foresee the student's final outcome in a course. Thus, enabling them to conduct in-time interventions towards to prevent underachievement. Many specialized EDM tools have been proposed to the EDM community, in order to compensate the shortcomings caused by general purpose DM tools. However, this review has shown that the existing tools still not meet the EDM community requirements.

Distributed knowledge discovery technology appears as a solution to the limitations found in current EDM tools. Among the existing modern distributed knowledge discovery solutions, SOA-based solution is more suitable for the implementation of an EDM tool for computer-based education environments.

CHAPTER 3

3. Proposed Framework

3.1. Introduction

This chapter introduces the proposed architecture for the distributed knowledge discovery EDM system, named, EDM-framework, and the implementation of this architecture that consist in three major modules, namely MDPWS (Moodle Data Provider Web Service), WDMWS (Weka Data Mining Web Service) and SOEDM (Service-Oriented Educational Data Mining). The chapter ends by describing an implementation of a Moodle module for prediction of students' performance that relies on the models generated through the EDM-framework.

3.2. EDM-framework

This study proposed a new architecture for EDM, named EDM-framework, which is composed of three components/modules that cooperate with each other in order to perform a typical EDM task. The architecture has been designed based on three main principles: abstraction, extensibility, and interoperability. Abstraction is achieved by providing a higher level of representation that allows standardization of data and output models. Extensibility is achieved by allowing existing systems to add new resources such as data sources, and algorithms by simply advertising them to the system. Interoperability is achieved by adopting Web Services technologies.

Figure 3.1 illustrates the system architecture of the EDM-framework. The follows describe details of the related modules of the architecture:

- Data Provider (DP): This module provides an abstraction for data collection. Basically, it will collect the data from E-learning environments by means of Web services. These Web services will be implemented inside the E-learning systems and they will rely on predefined data templates. These templates will specify the data which must be sent to the service. The Web service will respond its requests with an abstract and well-structured XML dataset that will allow portability of data across different modules.
- DM Service (DMS): This module supplies data mining methods and techniques as web services. This will be achieved through encapsulation of data mining frameworks/tools into web services. Each data mining tool will supply a collection of data mining methods and techniques for data pre-processing, classification, association rules, clustering, and regression. XML files will be used to exchange messages between this module and others.
- Integration: Integration module is the core part of the architecture, it will be responsible for integrating all modules to execute EDM tasks. This module will registers both DP services and DM services in order to be able to discover resources (i.e. data or algorithms). Also, integration module will be responsible for facilitating the construction of knowledge discovery workflows, so that users can plan, store, and re-execute their workflows and also manage their results. In addition, integration module will also serve as a toolkit to develop modules for automating and facilitate all the EDM processes for users who are not expert in data mining, for example, wizard tools, recommendation engines, free-parameter DM algorithms, etc.

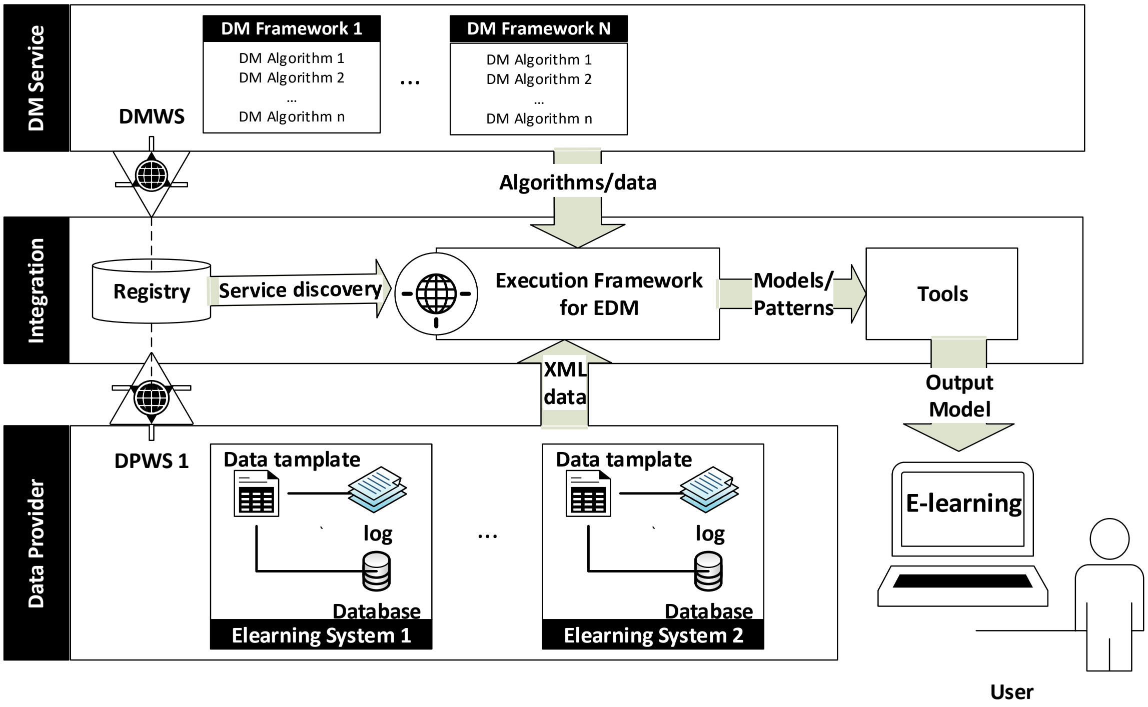


Figure 3.1: Architecture of the EDM-framework

3.2.1. EDM-framework development

The proposed architecture suggests a heterogeneous environment with complex interconnectivity. Therefore, this study has built the first prototype, which demonstrates the feasibility of the architecture. The prototype only includes libraries to support connection with Moodle LMS and Weka DM framework.

The architecture implementation was made up of three major modules. The first module provides an abstraction for data gathering, which was done by extending Moodle LMS source code. The second module provides DM methods and techniques as services, it was done by converting Weka API into a set of Web services. The third module acts as an intermediary between the first two modules, it contains a user-friendly interface that allows to dynamically locate data provider services, and run knowledge discovery tasks on data mining services. The system has been designed to be compatible, flexible and easy to modify.

The following sections provide the details of the implementation of each the modules mentioned above.

3.2.2. Moodle Data Provider Web Service (MDPWS)

Moodle 2.7 (Modular Object Oriented Developmental Learning Environment) was used to implement Data Provider module due its vast popularity among EDM researches, also because it has an open-source modular design that easily accommodates the addition of new functionalities. Furthermore, Moodle allows the creation of customized Web services through the addition of plugins.

3.2.2.1. Protocols

Web services on Moodle are written in PHP language and can be delivered via the following protocols:

- Representational state transfer**,** or REST: it is relatively lightweight protocol. It uses GET/POST HTTP methods on its requests and deliver the result in XML or JSON (JavaScript Object Notation) format ("Moodle," 2015).
- Simple Object Access Protocol, or SOAP: it is an HTTP messages exchange protocol, normally uses an XML-based file for describing Web services known as WSDL file. SOAP in Moodle is based on Zend framework ("Zend Framework," 2015), but its current implementation does not generate WSDL compatible with JAVA and DOT NET ("Moodle," 2015).

- XML-based Remote Procedure Call, or XML-RPC: It also based on Zend framework, it is similar to SOAP, however, is relatively lightweight and does not support WSDL ("Moodle," 2015).
- Action Message Format, or AMF: It based on Zend AMF and provides support for Adobe's Action Message format to allow communication between Moodle and Flash player or Flex applications ("Moodle," 2015).

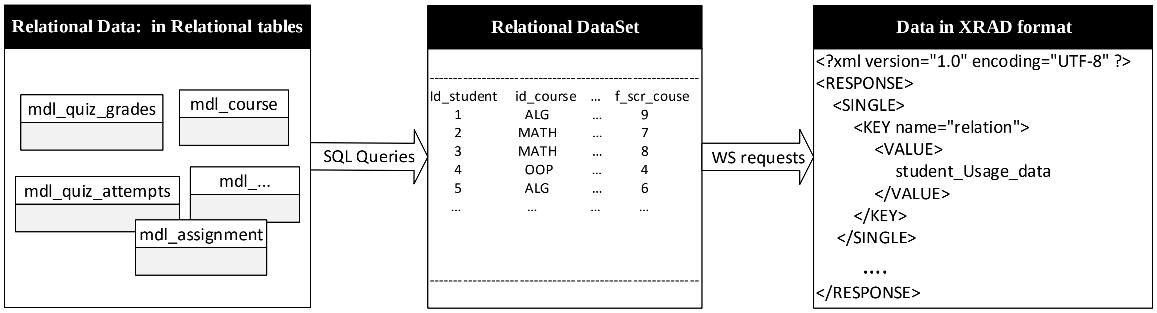
It was chosen REST protocol to support the implementation of MDPWS because it simplifies the communication between the MDPWS and its invokers.

3.2.2.2. Security

During service invocation, invokers must send a security key in order to be allowed to access the service. This key-based authentication mechanism secures the communication between invokers and Web service provider within a network.

3.2.2.3. XML "Relation-Attribute-Data" format

The ultimate goal of the MDPWS is to query the data from Moodle system and return it as an abstract and well-structured XML dataset. A dataset consists of a file/table with one or more record(s), where each record represents a transaction (Cristóbal Romero et al., 2014). Typically, a transaction includes a unique transaction ID (identity number) and a list of the items making up the transaction (Cristóbal Romero et al., 2014). Moodle does not provide directly any transactional database or dataset in itself (Cristóbal Romero et al., 2014), and usually stores its information in various relational database tables. Therefore, in order to deliver the Moodle's data as a dataset, this study has proposed a Moodle plugin that extends the Moodle Core Service and adds a new Web service for retrieving Moodle student's usage data. Essentially, what the proposed service does is to compile several database tables into a single summary table, then converts this summary table into well-structured XML notation and delivers it to the module who requested the service. To achieve this, the current study proposed an XML dataset implementation named “XML Relation-Attribute-Data” or simply XRAD that transforms the queries result set into an abstract XML dataset. Figure 3.2, illustrate how the data will flow from relational tables to XRAD dataset.



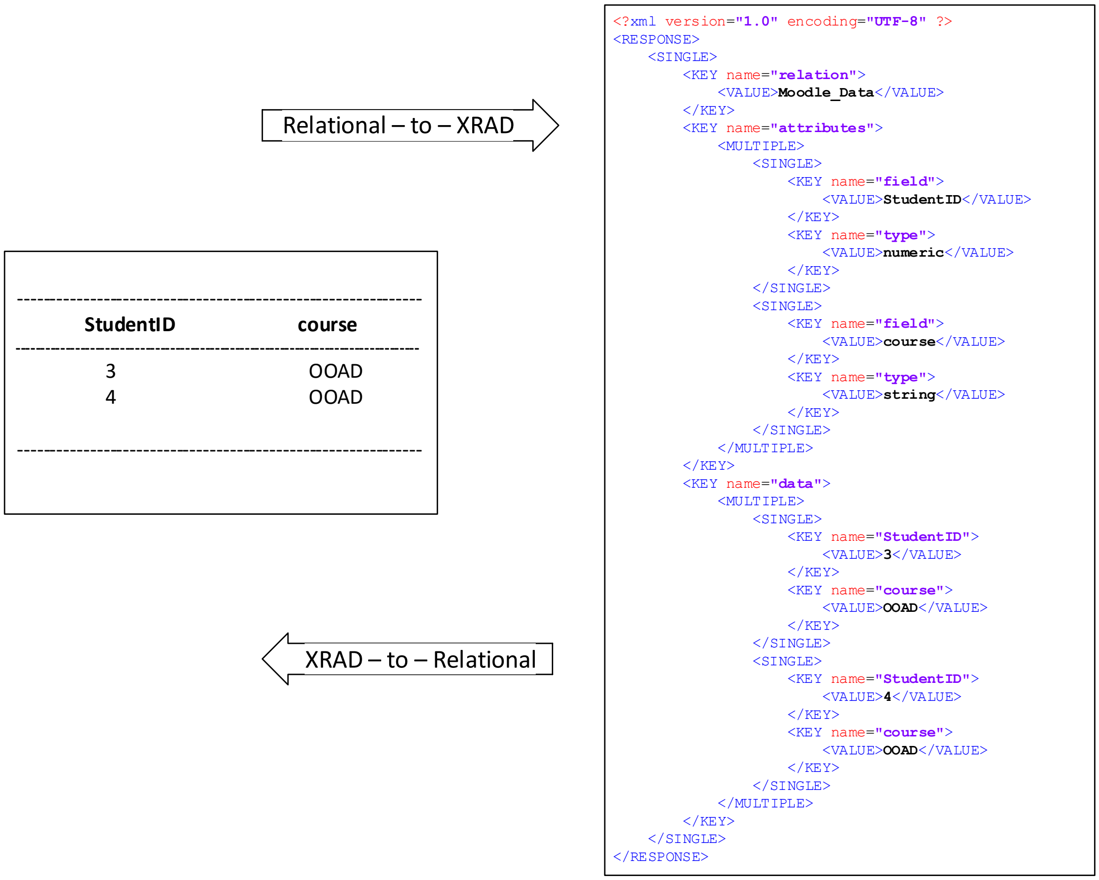
Id_student	id_course	...	f_scr_couse
1	ALG	...	9
2	MATH	...	7
3	MATH	...	8
4	OOP	...	4
5	ALG	...	6
...	...	...	...

```
<?xml version="1.0" encoding="UTF-8" ?>
<RESPONSE>
  <SINGLE>
     <KEY name="relation">
        <VALUE>
           student_Usage_data
        </VALUE>
     </KEY>
  </SINGLE>
     ....
</RESPONSE>
```

Figure 3.2: data flow from relational tables to XRAD dataset

With XRAD dataset notation the data collected from different E-learning environments can be structured into a generic data structure that will be understandable and portable across different modules. One benefit of this is, for instance, the process of transforming the data from XRAD to a specific DM format (i.e. ARFF, CSV) will be automatic. This will reduce drastically the effort, time, and resources needed for pre-processing, thereby decreasing the costs of EDM process.

In structure, XRAD consists of an XML file containing three parts: relation, attribute, and data. The relation section gives the name of the dataset. The attribute section describes the list of items making up the transaction along with the type of data contained in each item, and the data section contains the instance declarations lines. XRAD attributes support four kinds of data types, namely numeric, nominal, string and date. Figure 3.3 demonstrates the conversion of a sample relational data into XRAD dataset and vice-versa.



StudentID	course
3	OOAD
4	OOAD

```
<?xml version="1.0" encoding="UTF-8" ?>
<RESPONSE>
    <SINGLE>
        <KEY name="relation">
            <VALUE>Moodle_Data</VALUE>
        </KEY>
        <KEY name="attributes">
            <MULTIPLE>
                <SINGLE>
                    <KEY name="field">
                        <VALUE>StudentID</VALUE>
                    </KEY>
                    <KEY name="type">
                        <VALUE>numeric</VALUE>
                    </KEY>
                </SINGLE>
                <SINGLE>
                    <KEY name="field">
                        <VALUE>course</VALUE>
                    </KEY>
                    <KEY name="type">
                        <VALUE>string</VALUE>
                    </KEY>
                </SINGLE>
            </MULTIPLE>
        </KEY>
        <KEY name="data">
            <MULTIPLE>
                <SINGLE>
                    <KEY name="StudentID">
                        <VALUE>3</VALUE>
                    </KEY>
                    <KEY name="course">
                        <VALUE>OOAD</VALUE>
                    </KEY>
                </SINGLE>
                <SINGLE>
                    <KEY name="StudentID">
                        <VALUE>4</VALUE>
                    </KEY>
                    <KEY name="course">
                        <VALUE>OOAD</VALUE>
                    </KEY>
                </SINGLE>
            </MULTIPLE>
        </KEY>
    </SINGLE>
</RESPONSE>
```

Figure 3.3: illustration of data conversion from relational table to XRAD format

3.2.2.4. Service implementation

The implemented service relies on student's usage template (see Table 3-1). Because, student's usage data was by far the most commonly used dataset in EDM researches and most of the traditional data mining techniques have been tested over this set of data (Cristóbal Romero et al., 2014).

Student's usage data consists of all information gathered from student's interaction with Moodle activities (e.g., forum, chat, email achieve, quizzes, and assignments) and the final mark in the course (Cristóbal Romero, Ventura, Espejo, & Hervás, 2008). Due to the nature of the data, the template used in this study consider "*student's usage data*" as the information related to quizzes and assignments done by the students (see Table 3-1).

Table 3-1: Students' usage data template

Name	Description
userid	Identification number of the student
courseid	Identification number of the course
courseshortname	Short identification of the course
coursefullname	Course full name
useremail	Student's email
userfirstname	Student's first name
userlastname	Student's last name
n_assign	Number of assignments taken
n_assign_fail	Number of assignments failed
n_assign_pass	Number of assignments passed
n_quiz	Number of quizzes taken
n_quiz_fail	Number of quizzes failed
n_quiz_pass	Number of quizzes passed
finalgrade	Final score of the student obtained in a course

Source: Adapted from (Cristóbal Romero et al., 2014)

The goal of the service was to retrieve Moodle student's usage data, and its implementation consisted in the five top-down steps depicted in Figure 3.4. The first four steps were carried out the service's implementation, whereas in the last step were enabled the service's security and accessibility.

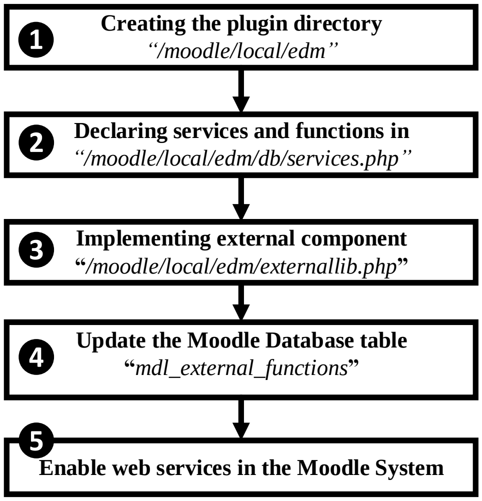

1
Creating the plugin directory
"/moodle/local/edm"
2
Declaring services and functions in
"/moodle/local/edm/db/services.php"
3
Implementing external component
"/moodle/local/edm/externallib.php"
4
Update the Moodle Database table
"mdl_external_functions"
5
Enable web services in the Moodle System

Figure 3.4: Service implementation workflow

The implementation of MDPWS has followed Moodle Web Services names and conventions. First of all, a plugin directory named "edm" was created in "/moodle/local/" directory. Next, a file was created in "/moodle/local/edm/db/services.php", where was declared the service and the function that implement the logic. Then, the implementation in terms of coding took place which was done by implementing three methods in a file located in "/moodle/local/edm/externallib.php" (see Figure 3.5). The first method implemented the logic for retrieving student's usage data (see Figure 3.5 line numbers 10-11, and Figure 3.5). The second method was used to describe the service's parameters (Figure 3.5 line numbers 12-13). The third method was used describe the service's return (Figure 3.5 line numbers 14-16). Two additional methods (*attributes_description* and *instances_description*) were also created to structure the service's return in order to deliver the data in XRAD format (Figure 3.5 line numbers 6-7).

```
<?php
  require_once($CFG->libdir . "/externallib.php");
  class local_edm_external extends external_api {

     // formatting return Key and Value
     public static function attributes_description() {...}
     public static function instances_description() {...}

     // function implementation
     public static function
          get_students_usage_data($param) {...}
     public static function
          get_students_usage_data_parameters(){...}
     public static function
          get_students_usage_data_returns(){...}
 }
```

Figure 3.5: externallib.php component class implementation

Figure 3.6 illustrate the pseudo-code used to implement the logic for retrieving student’s usage data. From a practical point of view, what the code does is to send a query to the database, and then format the result in order to deliver the data in XRAD format.

```
public static function get_students_usage_data($param = 'Default') {
     global $DB;
       // Capability checking and Implementation...Context validation,
       // relation - XML header
       $relation_name = 'students_usage_data';

       // SQL query for retrieve student’s usage data
      $sql = 'select * from mdl_students_usage_data_view';
      $users_data = $DB->get_recordset_sql($sql);
       // Attributes - XML Header
       $attributes = array(
           array('field' => 'id_student', 'type' => 'numeric'),
           // more attributes...
           array('field' => 'f_scr_course', 'type' => 'numeric')
       );

       // data -Retrieve users information.
       $data = array();
       foreach ($users_data as $user) {
           $userdetails['id_student'] = $user->id_student;
           // more data...
           $userdetails['f_scr_course'] = $user->f_scr_course;
           $data[] = $userdetails;
       }
       // Format return to deliver data in
       // relation-attribute-data structure
       return array('relation' => $relation_name,
                    'attribute' => $attributes,
                    'data' => $data);
   }
```

Figure 3.6: Logic for retrieving student’s usage data

The next section follows the implementation of DM service module, named WDMWS. The module WDMWS extended the well-known Weka toolkit to support remote execution of DM tasks.

3.2.3. Weka DM Web Service (WDMWS)

Weka (Waikato Environment for Knowledge Analysis) was used to implement DM Service module due its popularity among EDM researchers. Weka is a free and reliable tool which contains a collection of machine learning algorithms for pre-processing, classification, regression, clustering, association rules, and visualization (A. B. M. S. Ali

& Wasimi, 2007). Weka is an open-source data mining tool issued under GNU General Public License (GNU GPL) (Hall et al., 2009). Weka is currently distributed as a desktop Java application and can be downloaded from the official website (Vaghella, 2010). The WDMWS (Weka DM Web Service) module was written in Java language on top of Weka API for support remote execution of Weka DM algorithms by means of web services.

3.2.3.1. Architecture

The WDMWS architecture is based on the standard service-oriented life cycle (see Figure 3.7). Essentially, the WDMWS exposes a set of Web services operations– *xmlToArff*, *executeFilter*, *executeClassifier*, *executeCluster*, *executeAssociate*– that can be invoked remotely via a well-defined interface to execute DM tasks for pre-processing, classify, cluster and mining association rules. These services are defined using the Web Services Description Language (WSDL), published and stored in a Web services registry. The clients find and subscribe to the required Web services through the Web service registry. Once the clients are able to locate the Web services, they can request the remote execution of Web services operations.

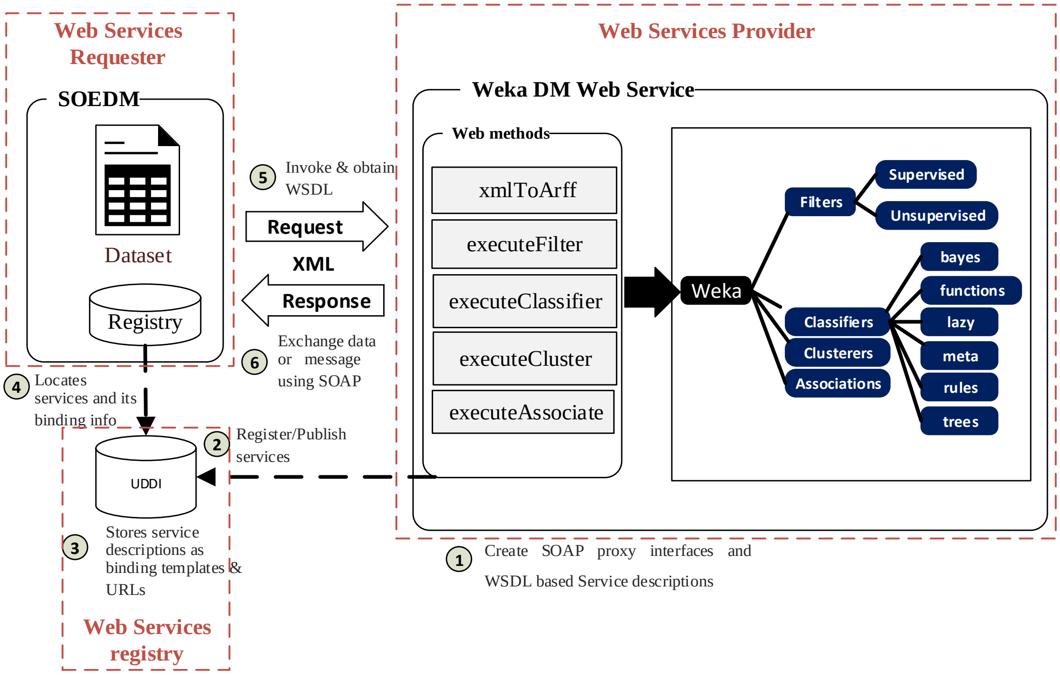


Figure 3.7: architecture of the Weka DM Web Service module

3.2.3.2. Web Services operations

There are five Web service operations currently implemented on the WDMWS, namely *xmlToArff*, *executeFilter*, *executeClassifier*, *executeCluster*, and *executeAssociate*. These operations expose all the pre-processing, classification, clustering and association rules algorithms provided by the Weka API.

The following tables provide a detailed description of each individual Web service operation along with an example of how its requests and response will take place. The first two operations describe operations for data pre-processing tasks, whereas the last three describe the operations for classification, clustering and association tasks, respectively.

Table 3-2: xmlToArff specifications

Operation signature	xmlToArff
Description	Performs conversion of an XRAD dataset into ARFF format.
Input	**xml** (String): training dataset in XML format.
Output	**xmlToArffResponse** (String): a dataset in ARFF format.

Example:

The following example shows how a sample an XRAD dataset (input) will be transformed into ARFF dataset (output) by means of *xmlToArff* web service method.

Table 3-3: Example of *xmlToArff*

Request	Input	`<?xml version="1.0" encoding="UTF-8" ?>`
		`<RESPONSE>`
		`    <SINGLE>`
		`        <KEY name="relation">`
		`            <VALUE>Moodle_Data</VALUE>`
		`        </KEY>`
		`        <KEY name="attributes">`
		`           …`
		`        </KEY>`
		`    </SINGLE>`
		`</RESPONSE>`

Response	Output	`@relation Moodle_Data`
		`@attribute StudentID numeric`
		`@attribute course string`
		`@data`
		`3, OOAD`
		`4, OOAD`

Table 3-4: executeFilter specifications

Operation signature	executeFilter
Description	Performs filtering on an ARFF dataset. The output is a filtered dataset in ARFF.
Input	1. **ArffTrainingDataset** (String): training dataset in ARFF format; 2. **Scheme** (String): filter scheme and options.
Output	**executeFilterResponse** (String): filtered dataset in ARFF format.

Example:

The following example shows how a sample an ARFF dataset (input 1) will be filtered by means of *executeFilter* web service method. The scheme (input 2) requests the removal of the first attribute (Remove –R 1), hence, resulting in a dataset with only one attribute (output).

Table 3-5: Example of *executeFilter*

Request	Input 1	`@relation Moodle_Data`
		`@attribute StudentID numeric`
		`@attribute course string`
		`@data`
		`3, OOAD`
		`4, OOAD`

	Input 2	`weka.filters.unsupervised.attribute.Remove -R 1`

Respons	Output	`@relation Moodle_Data`
		`@attribute course string`
		`@data`
		`OOAD`
		`OOAD`

Table 3-6: executeClassifier specifications

Operation signature	executeClassifier
Description	Submits the execution of a classification task by specifying the Scheme, model evaluation, and using ArffTrainingDataset as training dataset. By default, uses 10 folds cross-validation model evaluation.
Input	1. **ArffTrainingDataset** (String): training dataset in ARFF format; 2. **Scheme** (String): classification scheme and options; 3. **Evaluation** (String): evaluation options.
Output	**executeClassifierResponse** (String): Run information, classifier model (full training set), summary, detailed accuracy by class, and confusion matrix.

Example:

The following example shows how a sample an ARFF dataset (input 1) will be classified by means of *executeClassifier* web service method. The scheme (input 2) specifies the C4.5/J48 algorithm as the learning method using cross-validation as the evaluation method (input 3) by applying one random seed (- S 1) and two folds (- f 2). As a result (output), the service responded with a summary of information about the learning process along with the model.

Table 3-7: Example of *executeClassifier*

Request	Input 1	**@relation** Moodle_Data **@attribute** nAssiment {LOW,MEDIUM,HIGH} **@attribute** nQuizzPassed {LOW,MEDIUM,HIGH} **@attribute** mark {FAIL,PASS} **@data** LOW,HIGH, PASS MEDIUM,HIGH,FAIL LOW,MEDIUM,FAIL
	Input 2	weka.classifiers.trees.J48 -C 0.25 -M 2
	Input 3	cv –f 2 -S 1
Response	Output	=== Run information === Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2 Relation: Moodle_Data Instances: 3 Attributes: 3 nAssiment nQuizzPassed mark Test mode:2-fold cross-validation === Classifier model (full training set) === J48 pruned tree ------------------ : FAIL (3.0/1.0) Number of Leaves : 1 Size of the tree : 1 Time taken to build model: 0 seconds === Stratified cross-validation === === Summary === Correctly Classified Instances 2 66.6667% Incorrectly Classified Instances 1 33.3333% Kappa statistic 0 Mean absolute error 0.5 Root mean squared error 0.6455 Relative absolute error 100% Root relative squared error 124.5682% Total Number of Instances 3 === Detailed Accuracy By Class === TP Rate FP Rate Precision Recall F-Measure ROC Area Class 1 1 0.667 1 0.8 0.25 FAIL 0 0 0 0 0 0.25 PASS Weighted Avg. 0.667 0.667 0.444 0.667 0.533 0.25 === Confusion Matrix === a b <-- classified as 2 0 | a = FAIL 1 0 | b = PASS

Table 3-8: executeCluster specifications

Operation signature	executeCluster
Description	Submits the execution of a clustering task by specifying the Scheme, model evaluation, and using ArffTrainingDataset as training dataset. By default, uses 10 folds cross-validation model evaluation.
Input	1. **ArffTrainingDataset** (String): training dataset in ARFF format; 2. **Scheme** (String): clustering scheme and options; 3. **Evaluation** (String): evaluation options.
Output	**executeClusterResponse** (String): Run information, cluster, and measures of model accuracy.

Example:

The following example shows how a sample an ARFF dataset (input 1) will be clustered by means of *executeCluster* web service method. The scheme (input 2) specifies the KMeans algorithm as the learning method using percentage split as the evaluation method (input 3) by applying one random seed (- S 1). The data is split in 66% for training (- Z 66.0) and the remaining 34% for the test. As a result (output), the service responded with a summary of information about the learning process along with the clusters discovered from the data.

Table 3-9: Example *executeCluster*

<table>
<tr><td rowspan="3">Request</td><td>Input 1</td><td><pre>
@relation Moodle_Data
@attribute nAssiment {LOW,MEDIUM,HIGH}
@attribute nQuizzPassed {LOW,MEDIUM,HIGH}
@attribute mark {FAIL,PASS}
@data
LOW,HIGH, PASS
MEDIUM,HIGH,FAIL
LOW,MEDIUM,FAIL
</pre></td></tr>
<tr><td>Input 2</td><td><pre>weka.clusterers.SimpleKMeans</pre></td></tr>
<tr><td>Input 3</td><td><pre>p-split –Z 66.0 –S 1</pre></td></tr>
<tr><td>Response</td><td>Output</td><td><pre>
=== Run information ===
Scheme:weka.clusterers.SimpleKMeans
Relation: Moodle_Data
Instances: 3
Attributes: 3
              nAssiment
              nQuizzPassed
              mark
Test mode:split 66% train, remainder test
=== Model and evaluation on training set ==
kMeans
======
Number of iterations: 2
Within cluster sum of squared errors: 2.0
Missing values globally replaced with mean/mode
Cluster centroids:
                            Cluster#
Attribute   Full Data   0   1
                      (3)   (2)   (1)
===
nAssiment   LOW   LOW   LOW
nQuizzPassed   HIGH   HIGH   MEDIUM
mark   FAIL   FAIL   FAIL

Time taken to build model (full training data) : 0 seconds
=== Model and evaluation on test split ===
kMeans
======
Number of iterations: 1
Within cluster sum of squared errors: 0.0
Missing values globally replaced with mean/mode
Cluster centroids:
                            Cluster#
Attribute   Full Data   0
                      (1)   (1)
====================================
nAssiment   MEDIUM   MEDIUM
nQuizzPassed   HIGH   HIGH
mark   FAIL   FAIL
Time taken to build model (percentage split) : 0 seconds
Clustered Instances
0   2 (100%)
</pre></td></tr>
</table>

Table 3-10: executeAssociate specifications

Operation signature	executeAssociate
Description	Submits the execution of an association rules task by specifying the AssociatorScheme, and using ArffTrainingDataset as training dataset.
Input	1. **ArffTrainingDataset** (String): training dataset in ARFF format; 2. **Scheme** (String): association scheme and options.
Output	**executeAssociateResponse** (String): Run information, association rules, minimum support, and confidence.

Example:

The following example shows how a sample an ARFF dataset (input 1) will be mined to find relationships between different attributes by means of *executeAssociate* web service method. The scheme (input 2) specifies the Apriori algorithm as the learning by applying default options. As a result (output), the service responded with a summary of information about the learning process along with the discovered relationships in the form of rules.

Table 3-11: Example *executeAssociate*

Request	Input 1	`@relation Moodle_Data`
		`@attribute nAssiment {LOW,MEDIUM,HIGH}`
		`@attribute nQuizzPassed {LOW,MEDIUM,HIGH}`
		`@attribute mark {FAIL,PASS}`
		`@data`
		`LOW,HIGH, PASS`
		`MEDIUM,HIGH,FAIL`
		`LOW,MEDIUM,FAIL`

	Input 2	`weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1`

Response	Output	=== Run information === Scheme: weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 Relation: Moodle_Data Instances: 3 Attributes: 3 nAssiment nQuizzPassed mark === Associator model (full training set) === Apriori ======= Minimum support: 0.5 (1 instances) Minimum metric <confidence>: 0.9 Number of cycles performed: 10 Generated sets of large itemsets: Size of set of large itemsets L(1): 6 Size of set of large itemsets L(2): 9 Size of set of large itemsets L(3): 3 Best rules found: 1. nQuizzPassed=MEDIUM 1 ==> nAssiment=LOW 1 conf:(1) 2. mark=PASS 1 ==> nAssiment=LOW 1 conf:(1) 3. nAssiment=MEDIUM 1 ==> nQuizzPassed=HIGH 1 conf:(1) 4. nAssiment=MEDIUM 1 ==> mark=FAIL 1 conf:(1) 5. nQuizzPassed=MEDIUM 1 ==> mark=FAIL 1 conf:(1) 6. mark=PASS 1 ==> nQuizzPassed=HIGH 1 conf:(1) 7. nQuizzPassed=MEDIUM mark=FAIL 1 ==> nAssiment=LOW 1 conf:(1) 8. nAssiment=LOW mark=FAIL 1 ==> nQuizzPassed=MEDIUM 1 conf:(1) 9. nAssiment=LOW nQuizzPassed=MEDIUM 1 ==> mark=FAIL 1 conf:(1) 10. nQuizzPassed=MEDIUM 1 ==> nAssiment=LOW mark=FAIL 1 conf:(1)

As previously mentioned in section 3.3., the Integration module integrates all modules together. In other words, the Integration module acts like a middleware between the MDPWS and WDMWS to establish the remote execution of EDM tasks. The next section describes the implementation of Integration module.

3.2.4. SOEDM

SOEDM (Service-oriented EDM) is an implementation of Integration module of the EDM-framework. It is a desktop application written in java programming language and JavaFX framework. SOEDM act as a Web service client runtime environment which contains a Graphical User Interface (GUI) that users interact with it to remotely request the data from E-learning sources and then perform data mining tasks. During the process of requesting remote execution, the GUI manages each task independently by using asynchronous threads. This enables users to perform multiple DM tasks in parallel.

The current implementation of SOEDM comforts six functional requirements. Functional requirements specify the technical details of a system's behaviour and a common way to specify functional requirements is through building use-cases that describe the interactions between the system and external actors. An actor is a role that is played either by a person or by some other entity (Booch, Rumbaugh, Jacobson, & Matter, 1998). In the case of the EDM system, the actor can be an expert in EDM that wants to interact with the EDM system. Figure 3.8 provides a use-cases diagram that contains a set of use-cases for the system. Along with the use-cases diagram, follows tables that describe each individual use-case. Each use-case has a unique identifier (ID), along with a name, a description, pre-conditions (i.e. a list of condition that must be satisfied before the use-case starts), a summary of the interaction between use-case and actor (user).

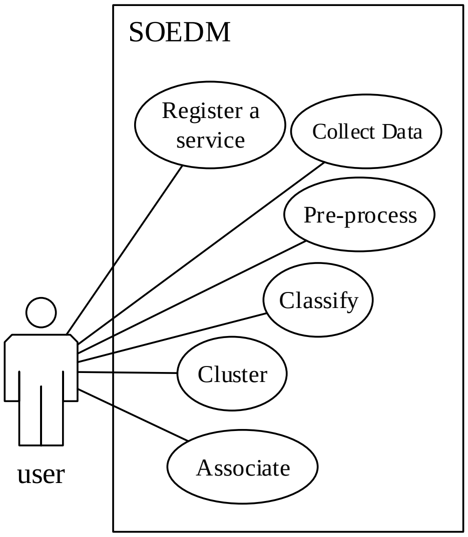


Figure 3.8: Use-cases diagram for distributed knowledge discovery

Table 3-12: Use-case UC-1: Register a service

ID	UC-1
Name	Register a service
Description	The user uses the system to register a Web service resource for connection with Data Provide or DM Service module.
Pre-condition	N/A
Summary	The system shall provide a way for user to: (1) Add a new connection; (2) Test an existing connection; (3) Edit an existing connection; (4) Delete an existing connection

Table 3-13: Use-case UC-2: Collect data

ID	UC-2
Name	Collect data
Description	The user uses the system to collect data from E-learning and then convert into a specific data mining format.
Pre-condition	Data provider Services and Data Mining Services connected.
Summary	The system shall provide a way to request data from Data Provider service, transform it into a specified data mining tool format, then display the dataset in the graphical user interface.

Table 3-14: Use-case UC-3: Pre-process

ID	UC-3
Name	Pre-process
Description	The user uses the system to prepare the data.
Pre-condition	Data collect from E-learning, and Data Mining Services connected.

Summary	The system shall provide a way to perform pre-processing tasks such as data cleansing, data transformation, data integration, data reduction, and data discretization.

Table 3-15: Use-case UC-4: Classify

ID	UC-4
Name	Classify
Description	The user uses the system to perform classification tasks.
Pre-condition	Data collect from E-learning, and Data Mining Services connected.
Summary	The system shall provide a way to build predictive models from the data collected on E-learning environment. In addition, the system must provide a mechanism to evaluate the model's performance.

Table 3-16: Use-case UC-5: Cluster

ID	UC-5
Name	Cluster
Description	The user uses the system to perform the clustering tasks.
Pre-condition	Data collect from E-learning, and Data Mining Services connected.
Summary	The system shall provide a way to learn clusters from the data collected on E-learning environment. In addition, the system must provide a mechanism to evaluate the model's performance.

Table 3-17: Use-case UC-6: Associate

ID	UC-6
Name	Associate
Description	The user uses the system to perform association task.
Pre-condition	Data collect for E-learning, and Data Mining Services connected.

Summary	The system shall provide a way to find association rules from the data collected on E-learning environment.

In addition, the SOEDM also comprises requirements related to the operation, quality and constraints of the EDM system which include: performance, scalability, reusability, extensibility, usability and maintainability.

3.2.4.1. Architecture

SOEDM system was developed based upon the architecture depicted in Figure 3.9. Essentially, what the SOEDM system does is to maintain a registry of XML files that stores information about the data sources connections schemes (e.g. Figure 3.11) and the algorithms schemes for filtering (e.g. Figure 3.10), classification (e.g. Figure 3.13), clusters (e.g. Figure 3.12) and association mining rules (e.g. Figure 3.14). The SOEDM system uses these XML files to support users in constructing requests for remote invocation. The SOEDM system uses a standard output pane to displays the output of a data mining tasks received from remote execution.

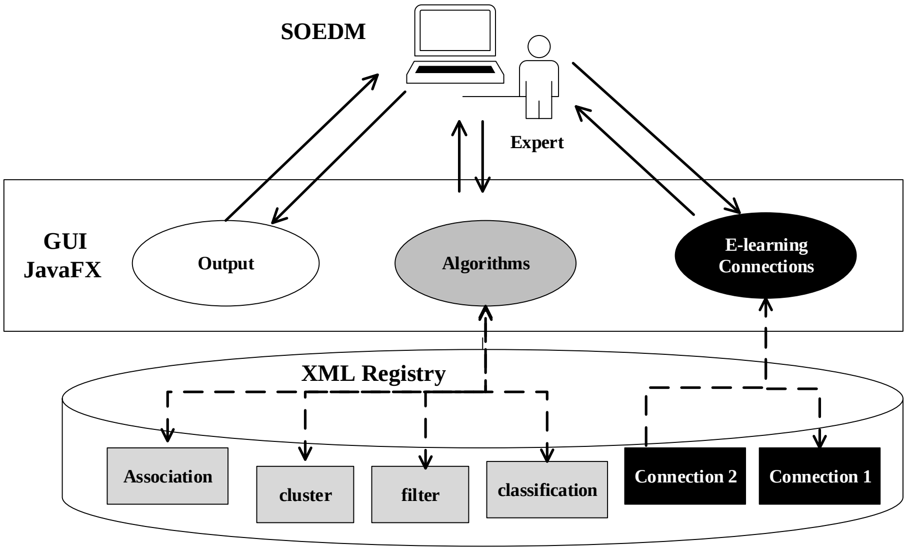
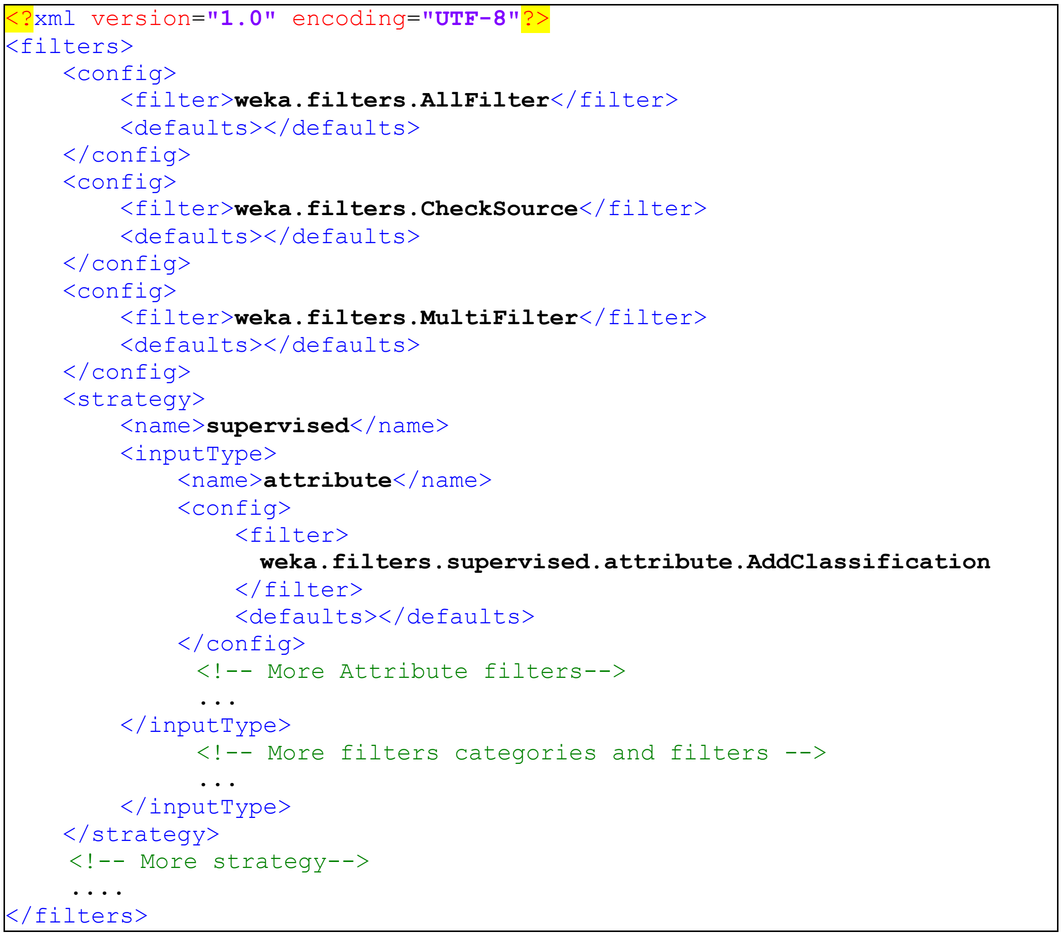
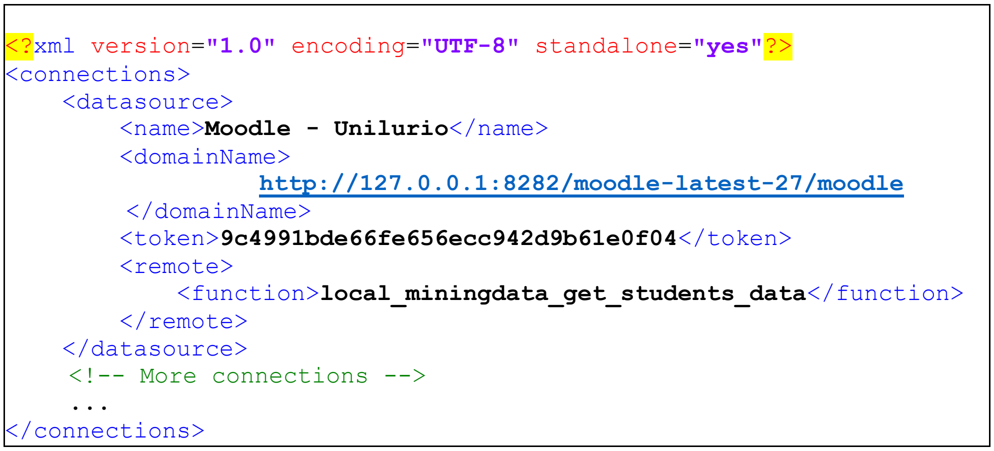


Figure 3.9: Architecture of the SOEDM module



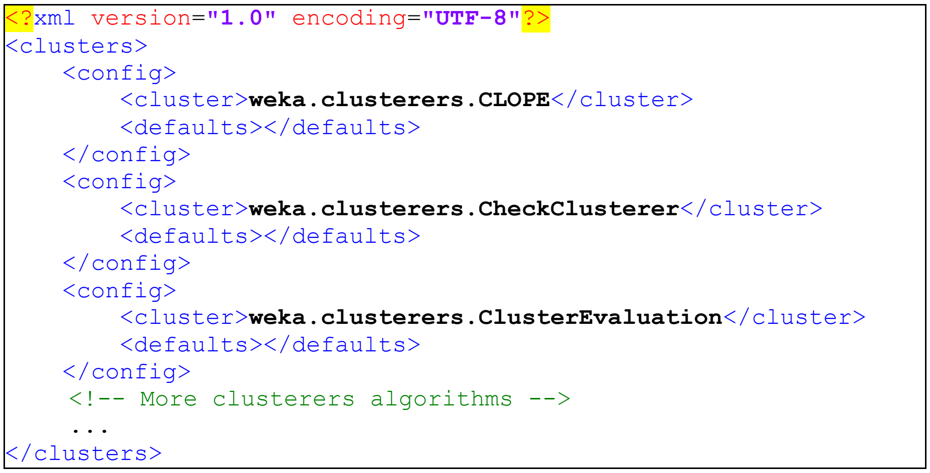
```xml
<?xml version="1.0" encoding="UTF-8"?>
<filters>
    <config>
        <filter>weka.filters.AllFilter</filter>
        <defaults></defaults>
    </config>
    <config>
        <filter>weka.filters.CheckSource</filter>
        <defaults></defaults>
    </config>
    <config>
        <filter>weka.filters.MultiFilter</filter>
        <defaults></defaults>
    </config>
    <strategy>
        <name>supervised</name>
        <inputType>
            <name>attribute</name>
            <config>
                <filter>
                  weka.filters.supervised.attribute.AddClassification
                </filter>
                <defaults></defaults>
            </config>
             <!-- More Attribute filters-->
             ...
        </inputType>
             <!-- More filters categories and filters -->
             ...
        </inputType>
    </strategy>
    <!-- More strategy-->
    ....
</filters>
```

Figure 3.10: XML for filtering schemes



```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<connections>
    <datasource>
        <name>Moodle - Unilurio</name>
        <domainName>
                  http://127.0.0.1:8282/moodle-latest-27/moodle
         </domainName>
        <token>9c4991bde66fe656ecc942d9b61e0f04</token>
        <remote>
            <function>local_miningdata_get_students_data</function>
        </remote>
    </datasource>
     <!-- More connections -->
     ...
</connections>
```

Figure 3.11: XML for connection



```xml
<?xml version="1.0" encoding="UTF-8"?>
<clusters>
    <config>
        <cluster>weka.clusterers.CLOPE</cluster>
        <defaults></defaults>
    </config>
    <config>
        <cluster>weka.clusterers.CheckClusterer</cluster>
        <defaults></defaults>
    </config>
    <config>
        <cluster>weka.clusterers.ClusterEvaluation</cluster>
        <defaults></defaults>
    </config>
     <!-- More clusterers algorithms -->
     ...
</clusters>
```

Figure 3.12: XML for clustering schemes

```xml
<?xml version="1.0" encoding="UTF-8"?>
<classifers>
    <method>
        <name>bayes</name>
        <config>
            <classifer>weka.classifiers.bayes.AODE</classifer>
            <defaults> </defaults>
        </config>
        <!-- More classifers -->
         ...
    </method>
    <method>
        <name>functions</name>
        <config>
            <classifer>
                      weka.classifiers.functions.GaussianProcesses
            </classifer>
            <defaults> </defaults>
        </config>
        <!-- More classifers -->
         ...
    </method>
     <!-- More classifers categories and classifers-->
     ...
</classifers>
```

Figure 3.13: XML for classification schemes

```xml
<?xml version="1.0" encoding="UTF-8"?>
<associations>
     <config>
        <associatior>weka.associations.Apriori</associatior>
        <defaults>-N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1</defaults>
    </config>
     <config>
        <associatior>weka.associations.GeneralizedSequentialPatterns</associatior>
        <defaults>-S 0.9 -I 0 -F -1</defaults>
    </config>
     <!-- More algorithms -->
    ...
</associations>
```

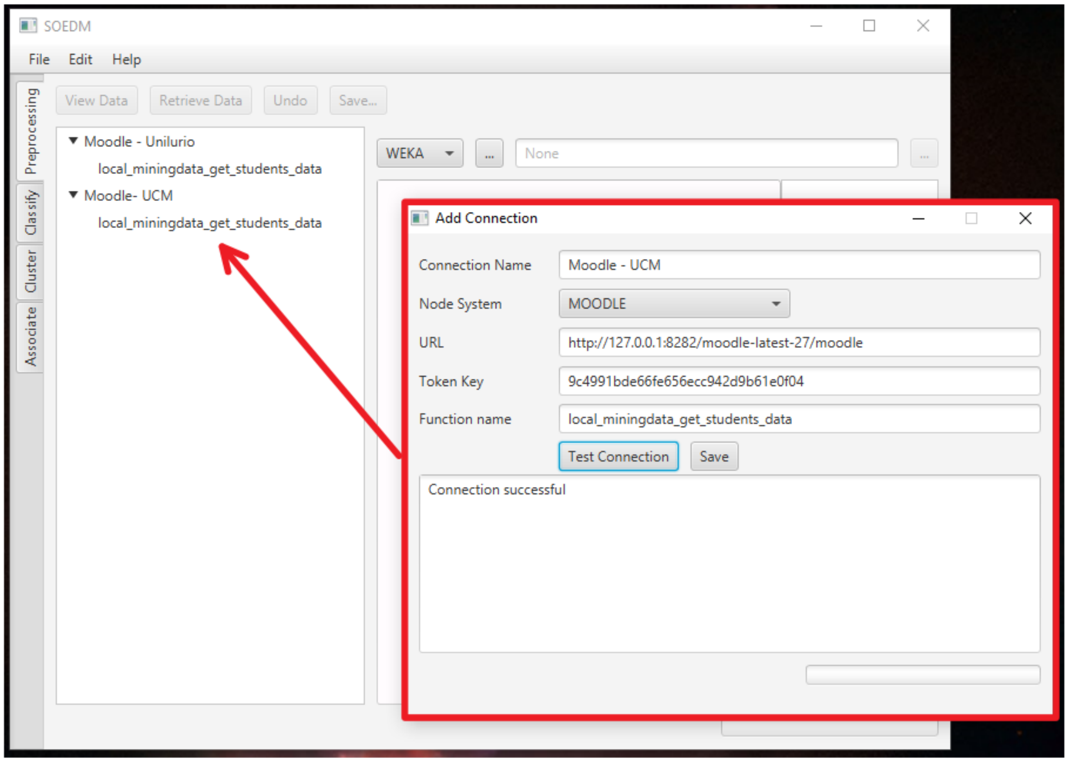
Figure 3.14: XML for association schemes

The SOEDM GUI is very similar to the Weka Explorer environment which includes pre-processing, classify, cluster and associate functionalities. Each of these functionalities can be easily accessed by activating a tab locate at the top left corner of the main GUI. In the next sections follows the description of each individual functionality of the SOEDM system.

3.2.4.2. Pre-processing

The pre-processing tab contains the implementations of the following uses cases:

- UC-1: Register a service. This use-case manages the connections for Web services resources. For instance, the Figure 3.15 shows how a connection to a Moodle Web service resource is added in the SOEDM application.
- UC-2: Collect data. This use-case aims at collecting data from E-learning resources. The Figure 3.16 shows how SOEDM retrieves the data from a given Moodle connection.
- UC-3: Pre-process or in other words, filtering the data. This use-case describes the transformation of the data in various ways. The Figure 3.17 shows how the SOEDM is used to filter the data using the *Remove* filter.



Connection Name: Moodle - UCM

Node System: MOODLE

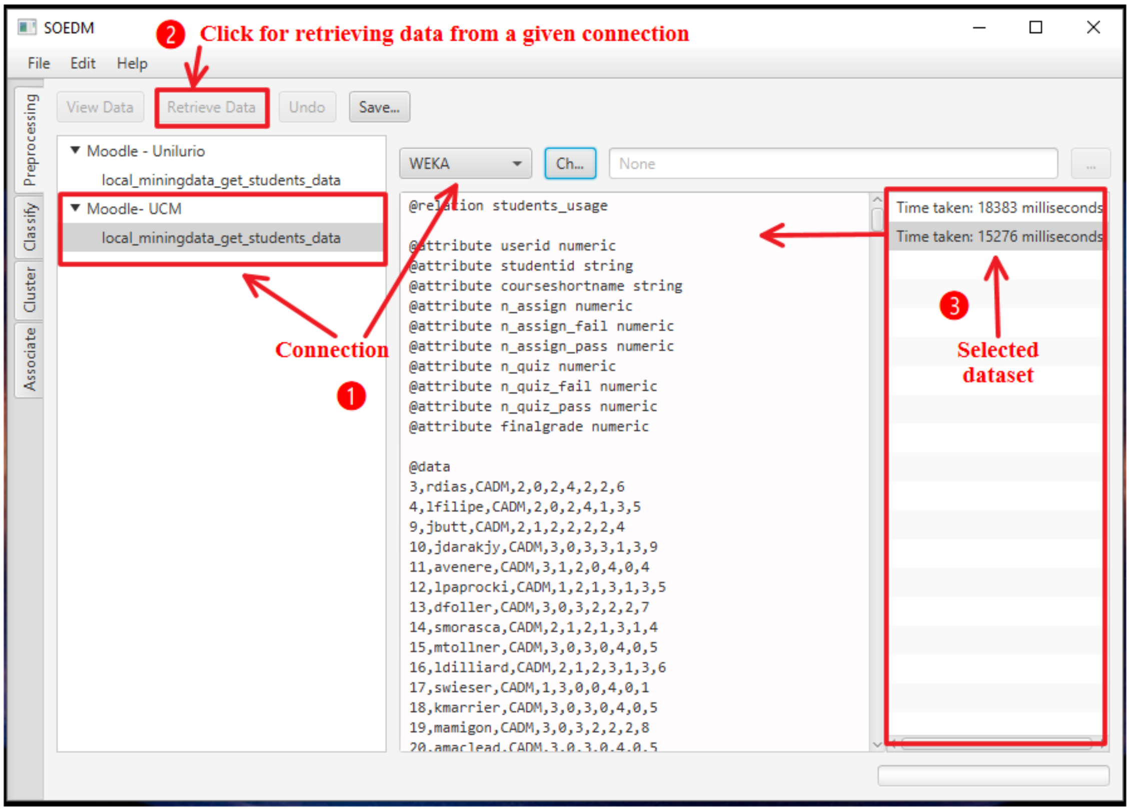
URL: http://127.0.0.1:8282/moodle-latest-27/moodle

Token Key: 9c4991bde66fe656ecc942d9b61e0f04

Function name: local_miningdata_get_students_data

Connection successful

Figure 3.15: Add Connection through pre-processing GUI

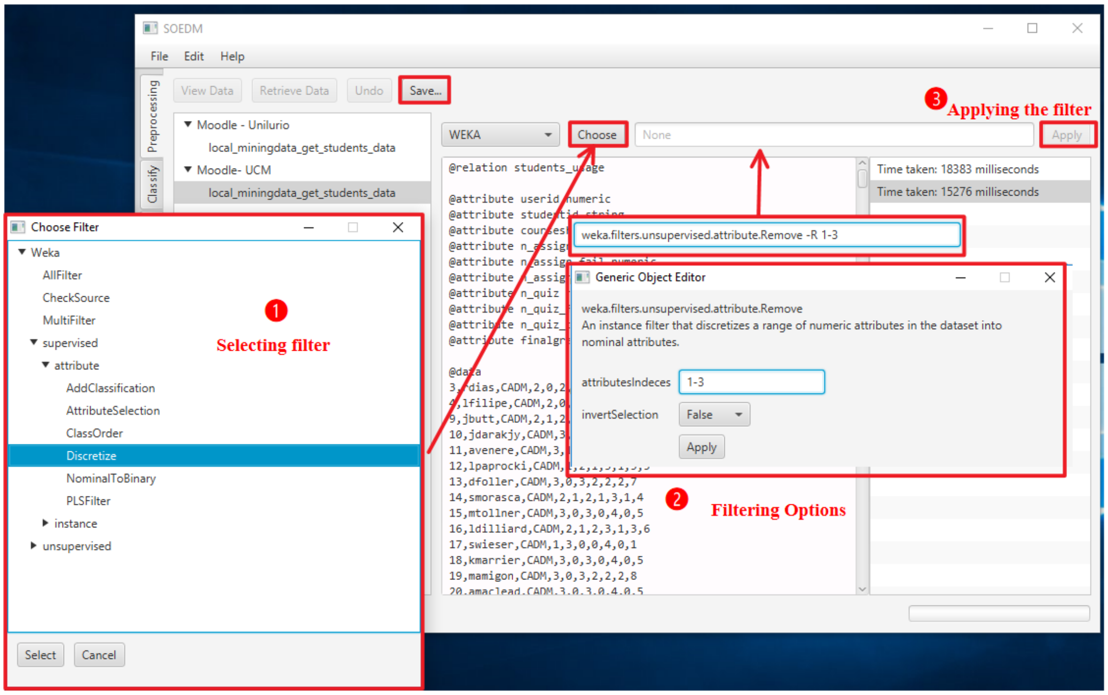


Time taken: 18383 milliseconds

Time taken: 15276 milliseconds

```
@relation students_usage

@attribute userid numeric
@attribute studentid string
@attribute courseshortname string
@attribute n_assign numeric
@attribute n_assign_fail numeric
@attribute n_assign_pass numeric
@attribute n_quiz numeric
@attribute n_quiz_fail numeric
@attribute n_quiz_pass numeric
@attribute finalgrade numeric

@data
3,rdias,CADM,2,0,2,4,2,2,6
4,lfilipe,CADM,2,0,2,4,1,3,5
9,jbutt,CADM,2,1,2,2,2,2,4
10,jdarakjy,CADM,3,0,3,3,1,3,9
11,avenere,CADM,3,1,2,0,4,0,4
12,lpaprocki,CADM,1,2,1,3,1,3,5
13,dfoller,CADM,3,0,3,2,2,2,7
14,smorasca,CADM,2,1,2,1,3,1,4
15,mtollner,CADM,3,0,3,0,4,0,5
16,ldilliard,CADM,2,1,2,3,1,3,6
17,swieser,CADM,1,3,0,0,4,0,1
18,kmarrier,CADM,3,0,3,0,4,0,5
19,mamigon,CADM,3,0,3,2,2,2,8
20,amaclead,CADM,3,0,3,0,4,0,5
```

Figure 3.16: Retrieving data from a Moodle LMS



```
@relation students_usage

@attribute userid numeric
@attribute studentid string
@attribute coursesh
@attribute n_assign
@attribute n_assign_fail numeric
@attribute n_assig
@attribute n_quiz
@attribute n_quiz_
@attribute n_quiz_
@attribute finalgr

@data
3,rdias,CADM,2,0,2
4,lfilipe,CADM,2,0
9,jbutt,CADM,2,1,2
10,jdarakjy,CADM,3
11,avenere,CADM,3,
12,lpaprocki,CADM,
13,dfoller,CADM,3,0,3,2,2,2,7
14,smorasca,CADM,2,1,2,1,3,1,4
15,mtollner,CADM,3,0,3,0,4,0,5
16,ldilliard,CADM,2,1,2,3,1,3,6
17,swieser,CADM,1,3,0,0,4,0,1
18,kmarrier,CADM,3,0,3,0,4,0,5
19,mamigon,CADM,3,0,3,2,2,2,8
20,amaclead,CADM,3,0,3,0,4,0,5
```

```
Time taken: 18383 milliseconds
Time taken: 15276 milliseconds
```

weka.filters.unsupervised.attribute.Remove -R 1-3

Generic Object Editor

weka.filters.unsupervised.attribute.Remove
An instance filter that discretizes a range of numeric attributes in the dataset into nominal attributes.

attributesIndeces: 1-3

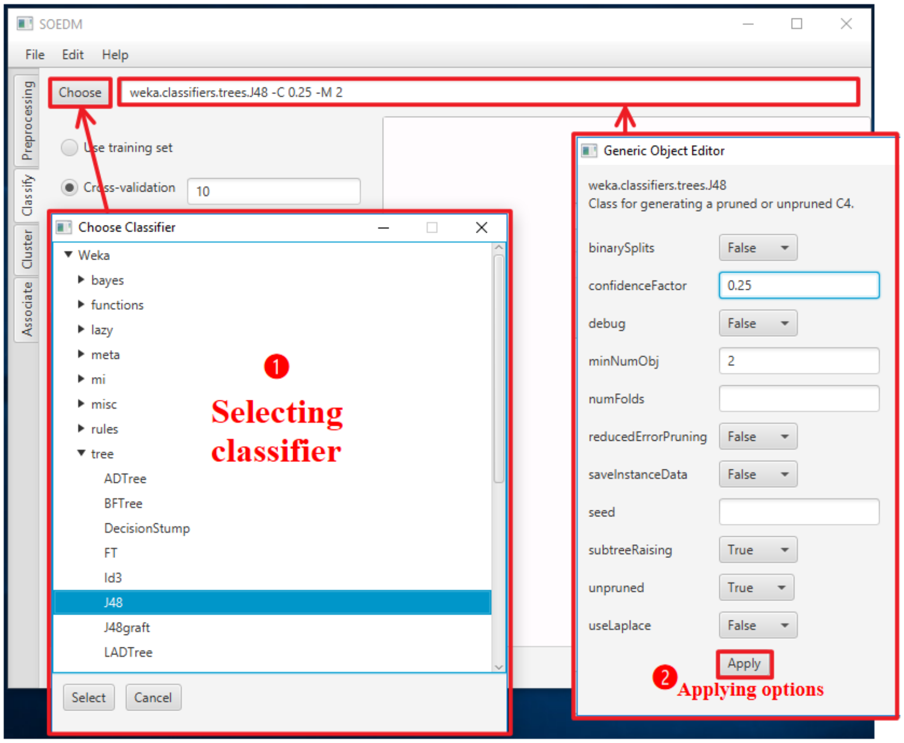
invertSelection: False

Apply

Figure 3.17: Filtering the data

3.2.4.3. Classification

The classify tab contains the implementation for the use-case UC-4-Classify. Through this GUI, users can select a classifier, select the test options (i.e. use training set, cross-validation or percentage split), and start a remote execution of classification task (see Figure 3.18 and Figure 3.19). The output of the classification task received from remote execution is then displayed in a standard output pane (Figure 3.19).



weka.classifiers.trees.J48 -C 0.25 -M 2

○ Use training set

● Cross-validation 10

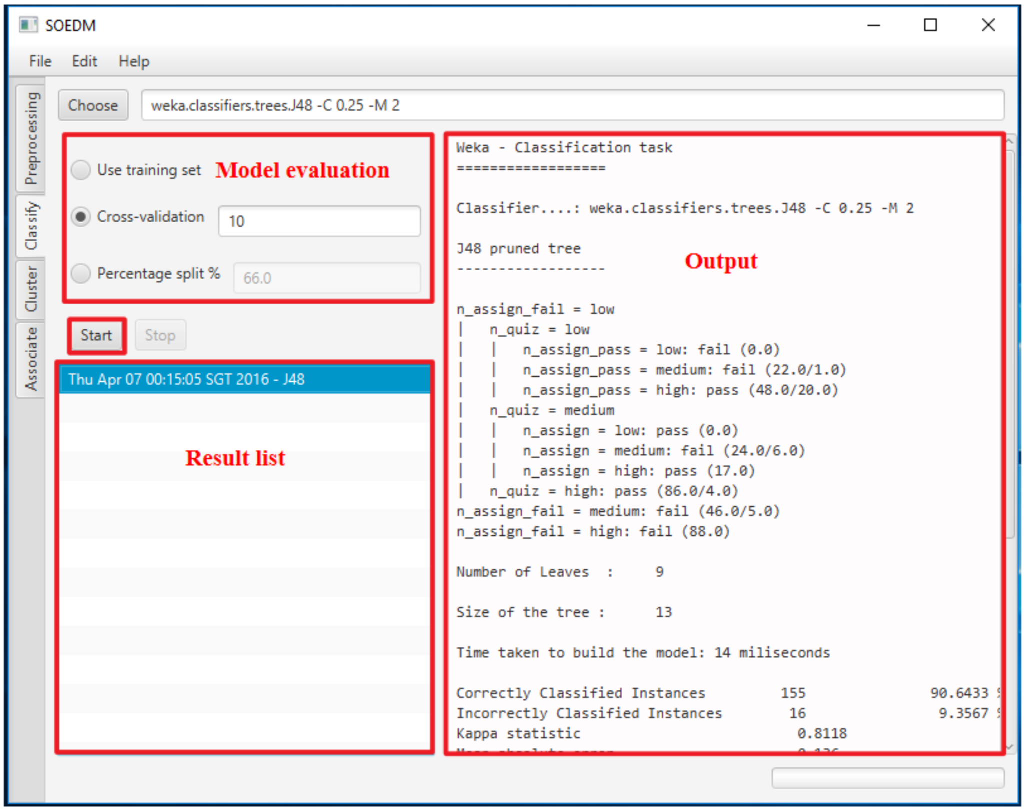
1 Selecting classifier

weka.classifiers.trees.J48
Class for generating a pruned or unpruned C4.

binarySplits	False
confidenceFactor	0.25
debug	False
minNumObj	2
numFolds	
reducedErrorPruning	False
saveInstanceData	False
seed	
subtreeRaising	True
unpruned	True
useLaplace	False

2 Applying options

Figure 3.18: Selecting classifier on classify tab



weka.classifiers.trees.J48 -C 0.25 -M 2

○ Use training set
● Cross-validation 10
○ Percentage split % 66.0

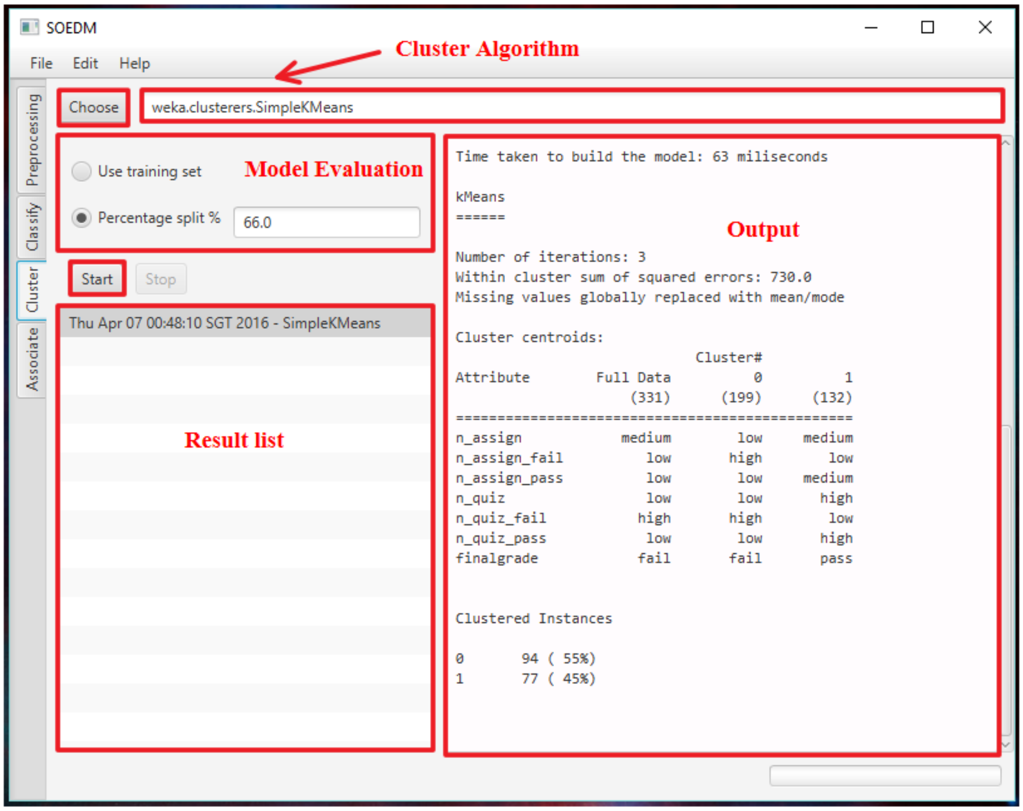
Thu Apr 07 00:15:05 SGT 2016 - J48

```
Weka - Classification task
==================

Classifier....: weka.classifiers.trees.J48 -C 0.25 -M 2

J48 pruned tree
------------------

n_assign_fail = low
|   n_quiz = low
|   |   n_assign_pass = low: fail (0.0)
|   |   n_assign_pass = medium: fail (22.0/1.0)
|   |   n_assign_pass = high: pass (48.0/20.0)
|   n_quiz = medium
|   |   n_assign = low: pass (0.0)
|   |   n_assign = medium: fail (24.0/6.0)
|   |   n_assign = high: pass (17.0)
|   n_quiz = high: pass (86.0/4.0)
n_assign_fail = medium: fail (46.0/5.0)
n_assign_fail = high: fail (88.0)

Number of Leaves  :     9

Size of the tree :      13

Time taken to build the model: 14 miliseconds

Correctly Classified Instances         155               90.6433
Incorrectly Classified Instances        16                9.3567
Kappa statistic                          0.8118
```

Figure 3.19: Output after a remote execution of classification task

3.2.4.4. Clustering

The cluster tab contains the implementation for the use-case UC-5-Cluster. Through this GUI, users can select a cluster, select the test options (i.e. use training set or percentage split), and start a remote execution of clustering task. The output of the clustering task received from remote execution is then displayed in a standard output pane (Figure 3.20).



weka.clusterers.SimpleKMeans

Percentage split % 66.0

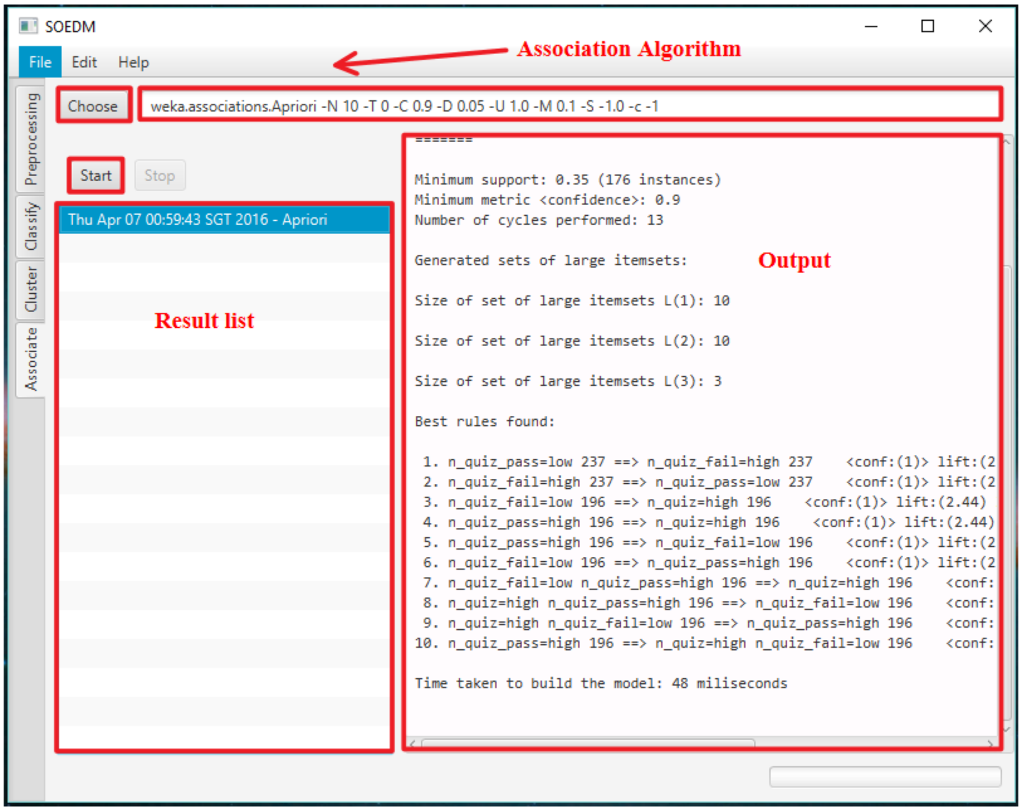
Thu Apr 07 00:48:10 SGT 2016 - SimpleKMeans

```
Time taken to build the model: 63 miliseconds

kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 730.0
Missing values globally replaced with mean/mode

Cluster centroids:
                             Cluster#
Attribute        Full Data          0          1
                     (331)      (199)      (132)
================================================
n_assign            medium        low     medium
n_assign_fail          low       high        low
n_assign_pass          low        low     medium
n_quiz                 low        low       high
n_quiz_fail           high       high        low
n_quiz_pass            low        low       high
finalgrade            fail       fail       pass


Clustered Instances

0       94 ( 55%)
1       77 ( 45%)
```

Figure 3.20: Output after a remote execution of clustering task

3.2.4.5. Associating

The associate tab contains the implementation for the use-case UC-6-Associate. Through this GUI, users can select the association mining algorithm and start a remote execution of clustering task. The output of the association task received from remote execution is then displayed in a standard output pane (Figure 3.21).



weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

Thu Apr 07 00:59:43 SGT 2016 - Apriori

```
=======

Minimum support: 0.35 (176 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 13

Generated sets of large itemsets:

Size of set of large itemsets L(1): 10

Size of set of large itemsets L(2): 10

Size of set of large itemsets L(3): 3

Best rules found:

 1. n_quiz_pass=low 237 ==> n_quiz_fail=high 237    <conf:(1)> lift:(2
 2. n_quiz_fail=high 237 ==> n_quiz_pass=low 237    <conf:(1)> lift:(2
 3. n_quiz_fail=low 196 ==> n_quiz=high 196    <conf:(1)> lift:(2.44)
 4. n_quiz_pass=high 196 ==> n_quiz=high 196    <conf:(1)> lift:(2.44)
 5. n_quiz_pass=high 196 ==> n_quiz_fail=low 196    <conf:(1)> lift:(2
 6. n_quiz_fail=low 196 ==> n_quiz_pass=high 196    <conf:(1)> lift:(2
 7. n_quiz_fail=low n_quiz_pass=high 196 ==> n_quiz=high 196    <conf:
 8. n_quiz=high n_quiz_pass=high 196 ==> n_quiz_fail=low 196    <conf:
 9. n_quiz=high n_quiz_fail=low 196 ==> n_quiz_pass=high 196    <conf:
10. n_quiz_pass=high 196 ==> n_quiz=high n_quiz_fail=low 196    <conf:

Time taken to build the model: 48 miliseconds
```

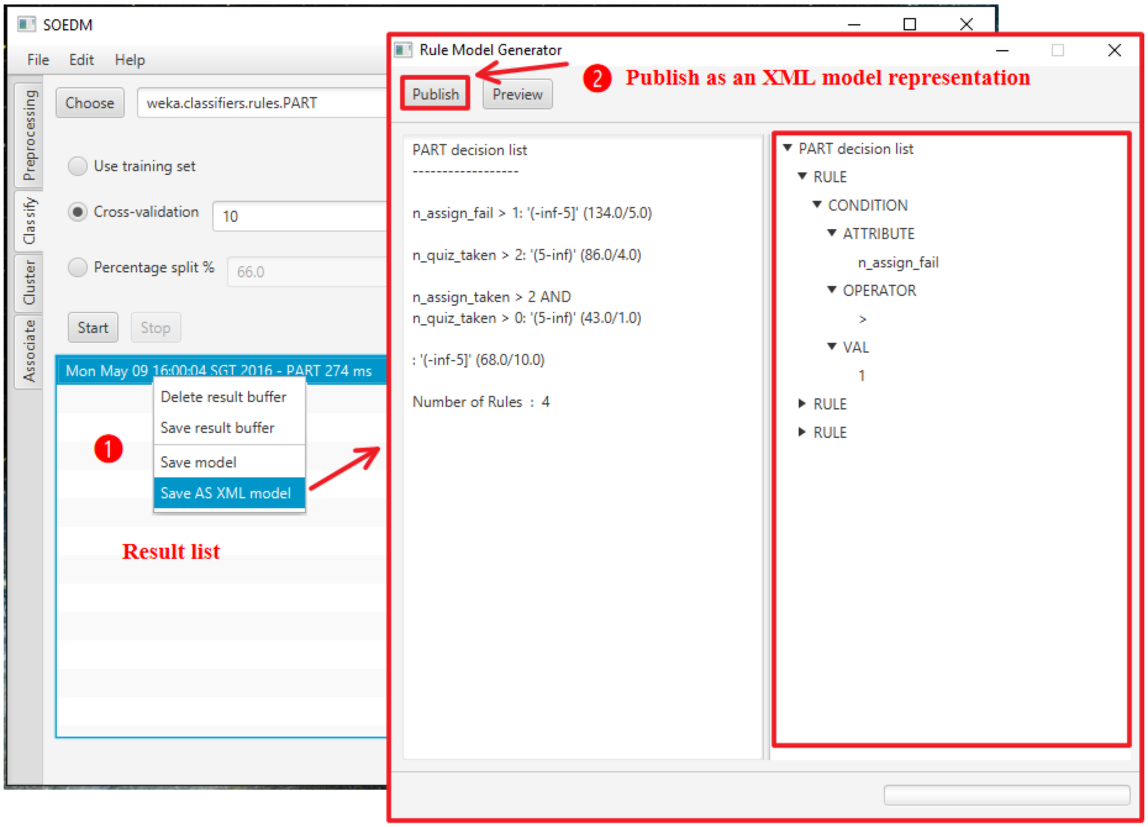
Figure 3.21: Output after a remote execution of association task

Apart from the above functionalities, SOEDM also includes features for saving datasets and models. As an example, the Figure 3.17 shows the Save… button at the top of the preprocess tab. This button can be used to save the currently selected dataset in file formats that can represent the relation, hence allowing it to be kept for future use.

In addition, the result list generated after execution of a DM tasks (classification, cluster or association) can be managed by right-clicking an entry which invokes a menu containing these items (Figure 3.22):

1. Delete result buffer. It removes a selected entry from the results.

2. Save result buffer. Brings up a dialog allowing users to save the text file containing the textual output.
3. Save model. Saves a model object to a binary file. Objects are saved in Java ‘serialized object’ form.
4. Save As XML model. This item only appears when the model involved is built from PART algorithm. It brings up a dialog allowing users to structure the PART model and publish it as an XML file. This XML file will contain rules that can be fed into e-learning system knowledge base. The structure of the XML file is fairly simple and is shown in Figure 3.23. For instance, using student’s usage data template, the PART algorithm has uncovered 4 rules as shown in Figure 3.22. The first rule saying that if students fails more than 1 assignment(s) then s/he is likely to FAIL the course. The second rule saying that if students take more than 2 quizzes then s/he is likely to PASS the course. The third rule saying that if students take more than 2 assignments AND take at least 1 quiz then s/he is likely to PASS the course. And the last rule saying that if none of the above three rules occur then the student is likely to FAIL the course. The user can use the box at the right corner of the GUI to create and structure the rules and save as XML file.



weka.classifiers.rules.PART

○ Use training set

● Cross-validation 10

○ Percentage split % 66.0

Mon May 09 16:00:04 SGT 2016 - PART 274 ms

1

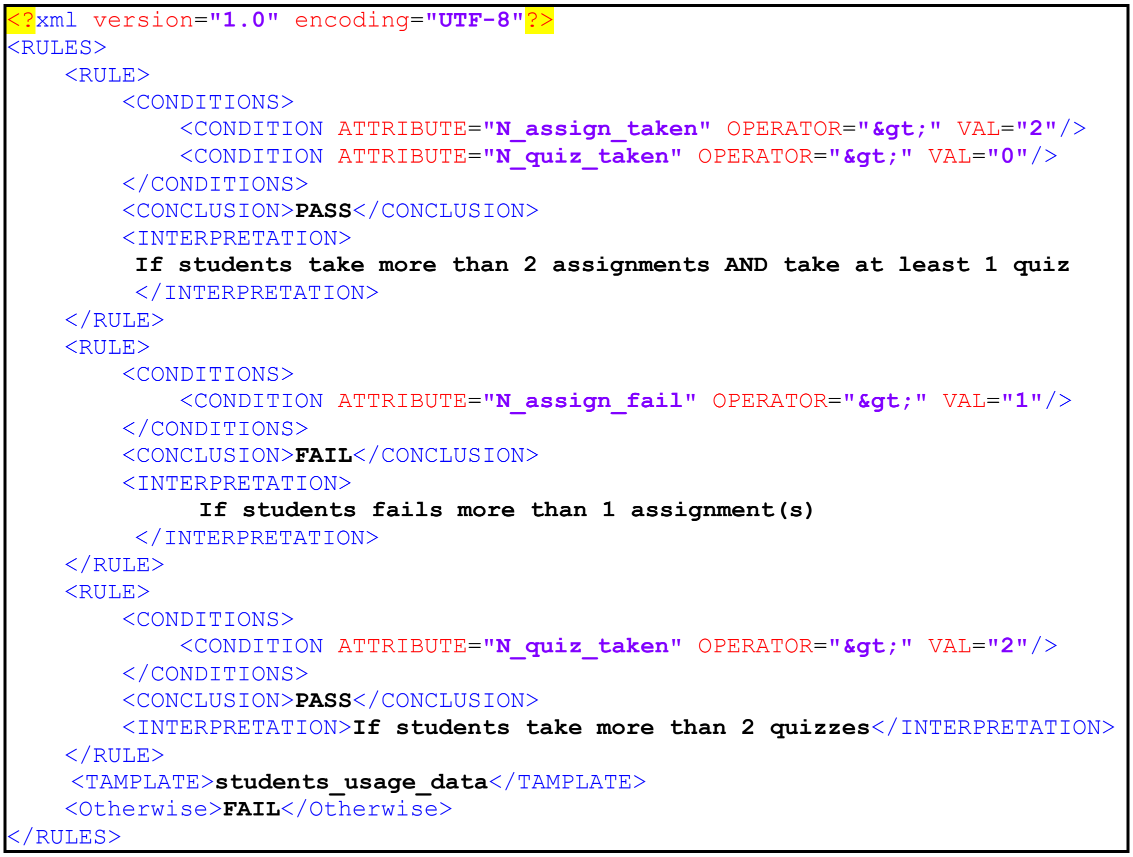
Result list

2 Publish as an XML model representation

```
PART decision list
------------------

n_assign_fail > 1: '(-inf-5]' (134.0/5.0)

n_quiz_taken > 2: '(5-inf)' (86.0/4.0)

n_assign_taken > 2 AND
n_quiz_taken > 0: '(5-inf)' (43.0/1.0)

: '(-inf-5]' (68.0/10.0)

Number of Rules  :  4
```

```
▼ PART decision list
  ▼ RULE
    ▼ CONDITION
      ▼ ATTRIBUTE
          n_assign_fail
      ▼ OPERATOR
          >
      ▼ VAL
          1
  ▶ RULE
  ▶ RULE
```

Figure 3.22: Saving the result as an XML rule-based model representation



```
<?xml version="1.0" encoding="UTF-8"?>
<RULES>
    <RULE>
        <CONDITIONS>
            <CONDITION ATTRIBUTE="N_assign_taken" OPERATOR="&gt;" VAL="2"/>
            <CONDITION ATTRIBUTE="N_quiz_taken" OPERATOR="&gt;" VAL="0"/>
        </CONDITIONS>
        <CONCLUSION>PASS</CONCLUSION>
        <INTERPRETATION>
         If students take more than 2 assignments AND take at least 1 quiz
         </INTERPRETATION>
    </RULE>
    <RULE>
        <CONDITIONS>
            <CONDITION ATTRIBUTE="N_assign_fail" OPERATOR="&gt;" VAL="1"/>
        </CONDITIONS>
        <CONCLUSION>FAIL</CONCLUSION>
        <INTERPRETATION>
             If students fails more than 1 assignment(s)
         </INTERPRETATION>
    </RULE>
    <RULE>
        <CONDITIONS>
            <CONDITION ATTRIBUTE="N_quiz_taken" OPERATOR="&gt;" VAL="2"/>
        </CONDITIONS>
        <CONCLUSION>PASS</CONCLUSION>
        <INTERPRETATION>If students take more than 2 quizzes</INTERPRETATION>
    </RULE>
     <TAMPLATE>students_usage_data</TAMPLATE>
    <Otherwise>FAIL</Otherwise>
</RULES>
```

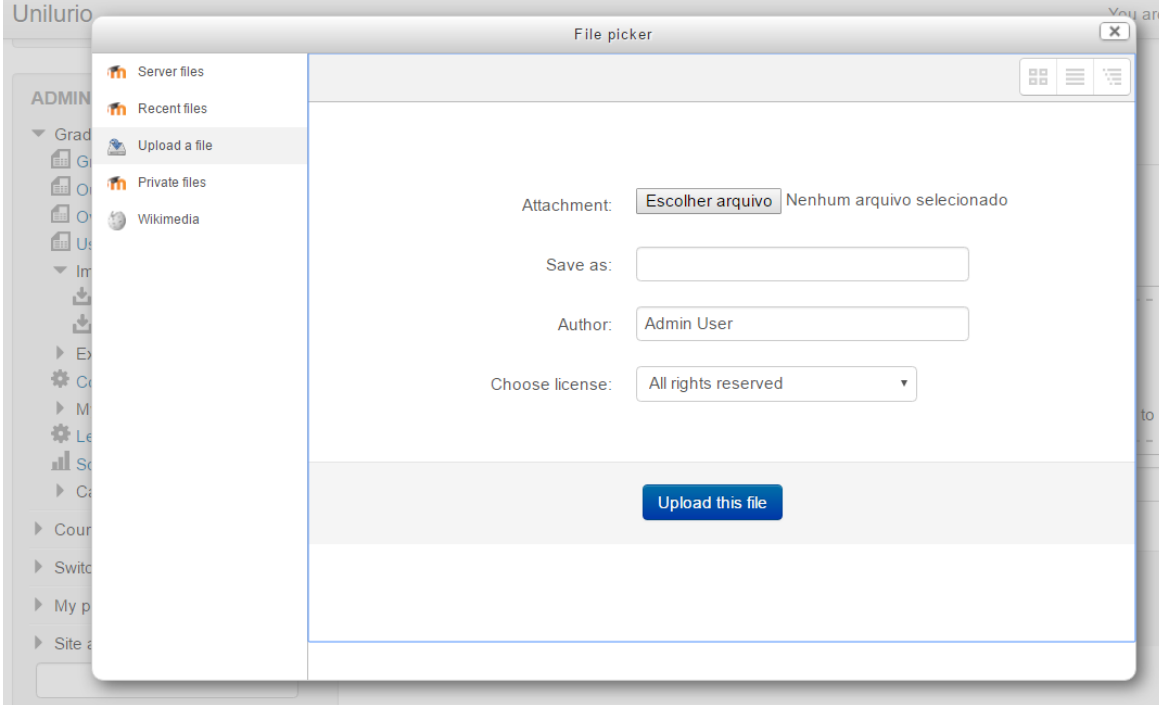
Figure 3.23: XML rule-based encoded patterns

In order to provide educators in Moodle environment with a report that goes beyond the simple static reports that are provided by Moodle, this study has proposed a Moodle module that extends the Moodle's report plugin and adds a new report for predicting student's performance. The next section describes the implementation of this module.

3.3. Student's performance prediction report

Student's performance prediction report allows educators in Moodle environment to browse through the list of students involved in their course and see the prediction of a success of each student (Figure 3.26) on that particular course. The predictions rely on the

XML rule-based model as the one depicted in Figure 3.23. These models can be built from SOEDM system, and then uploaded to the Moodle system by mean of the interface depicted in Figure 3.24.



Attachment: Escolher arquivo Nenhum arquivo selecionado

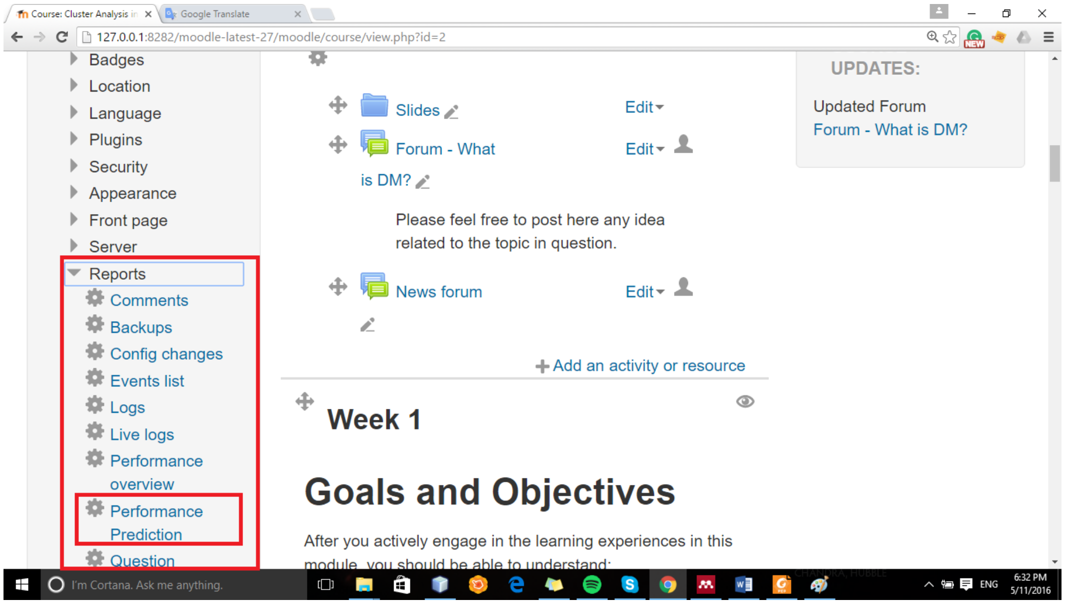
Save as:

Author: Admin User

Choose license: All rights reserved

Figure 3.24: Uploading XML rule-based model into Moodle system

The Student’s performance prediction module reads the XML rule-based model file and adds to its knowledge base. Then, the educators can navigate to student’s performance prediction report by selecting the link “Performance prediction” in the Moodle reporting block (Figure 3.25).



Please feel free to post here any idea related to the topic in question.

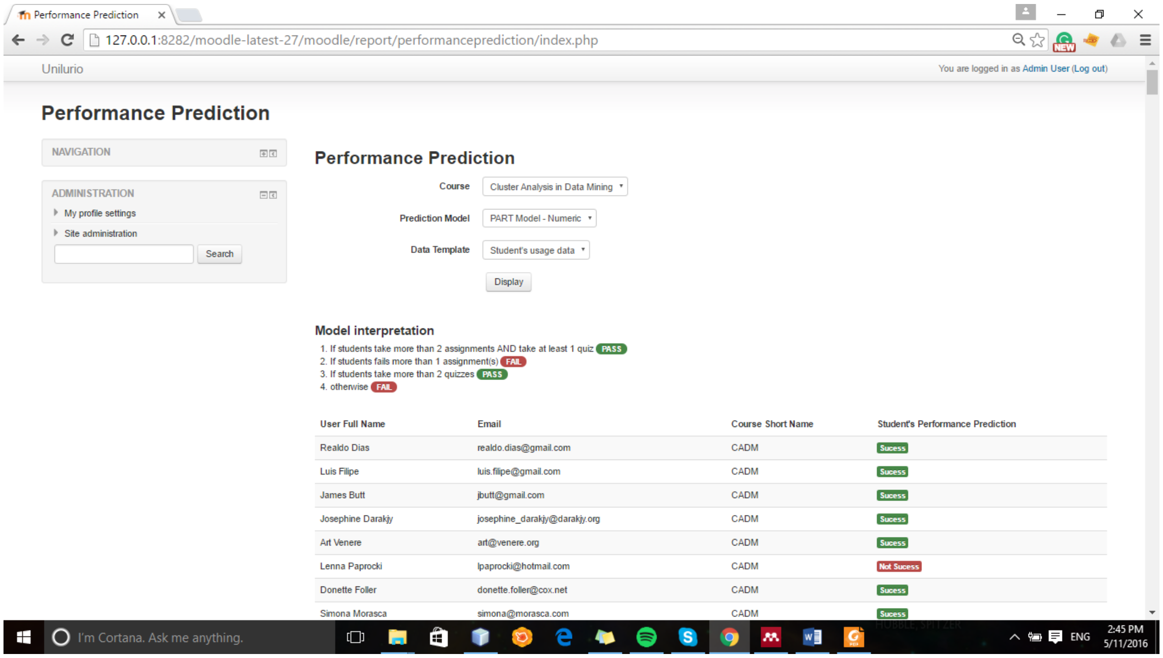
Week 1

Goals and Objectives

After you actively engage in the learning experiences in this module, you should be able to understand:

Figure 3.25: Moodle block for accessing performance prediction

Based on the report information, educators in online environment can act as soon as they become aware of a possible failures or underachievement in their courses. Therefore, enabling them to take proactive actions towards to prevent failures as well as to engage earlier with those students who are not at-risk in order to keep them on the right track.



Performance Prediction
127.0.0.1:8282/moodle-latest-27/moodle/report/performanceprediction/index.php
Unilurio
You are logged in as Admin User (Log out)
NAVIGATION
ADMINISTRATION
My profile settings
Site administration
Search
I'm Cortana. Ask me anything.
ENG
2:45 PM
5/11/2016

Performance Prediction

Performance Prediction

Course: Cluster Analysis in Data Mining

Prediction Model: PART Model - Numeric

Data Template: Student's usage data

Display

Model interpretation

1. If students take more than 2 assignments AND take at least 1 quiz PASS
2. If students fails more than 1 assignment(s) FAIL
3. If students take more than 2 quizzes PASS
4. otherwise FAIL

User Full Name	Email	Course Short Name	Student's Performance Prediction
Realdo Dias	realdo.dias@gmail.com	CADM	Sucess
Luis Filipe	luis.filipe@gmail.com	CADM	Sucess
James Butt	jbutt@gmail.com	CADM	Sucess
Josephine Darakjy	josephine_darakjy@darakjy.org	CADM	Sucess
Art Venere	art@venere.org	CADM	Sucess
Lenna Paprocki	lpaprocki@hotmail.com	CADM	Not Sucess
Donette Foller	donette.foller@cox.net	CADM	Sucess
Simona Morasca	simona@morasca.com	CADM	Sucess

Figure 3.26: New Moodle report for predicting student's performance

3.4. Summary

The chapter presents the proposed architecture for the distributed knowledge discovery EDM system as well as its implementation. The architecture, named EDM-framework, is made up of three major modules/components: Data Provider (DP), DM Service (DMS), and Integration.

The architecture has been designed based on three main principles: abstraction, extensibility, and interoperability. An implementation of the proposed architecture is presented to demonstrate the feasibility of the architecture. The implementation consisted in a three modules prototype that also includes a wizard for predicting student performance based on the student's usage data on Moodle environment. The first module, named MDPWS, provides an abstraction for data gathering, which was done by extending Moodle

LMS source code. The second module, named WDMWS, provides DM methods and techniques as services, it was done by converting Weka API into a set of Web services. The third module, named SOEDM, acts as an intermediary between the first two modules, it contains a user-friendly interface that allows to dynamically locate data provider services, and run knowledge discovery tasks on data mining services.

The chapter also presents a new Moodle report that enables educators in online environment to predict the students who are more likely to fail/success academically. The predictions are based upon the models generated by the SOEDM system.

CHAPTER 4

4. Methodology

4.1. Introduction

Two experiments were carried out in this study. The first experiment was conducted in order to select an accurate and comprehensible classifier for implementing the student's performance prediction module in Moodle system, while the second experiment was conducted to assess the overall performance of the proposed framework. The details of both experiments are given in the following sections.

4.2. Classifiers performance and comprehensibility evaluation

The goal of the first sets of experiments were to assess the quality of different classifiers for predicting student's final marks with respect to: the performance and the comprehensibility. This experiment was done as a part of development of the student's performance prediction module in Moodle (see section 3.3.). So that this feature could be built on top of comprehensible and accurate models that explains the system's predictions, in the way that users would not have difficulties to understand the system outputs.

The experiment seeks to find answer for the following question:

- What accurate and comprehensible model can be used for predicting student's final marks based on the students' usage data?

4.2.1. Classifiers performance metrics

There are many performance metrics for evaluating classification algorithms, and accuracy (the fraction of correctly classified instances) is a popular choice among EDM researchers. But there are cases where accuracy needs to be traded off with other measures in order to obtain better results. As an example, the case of imbalanced data, it is not appropriate to evaluate classifier's performance on the basis of accuracy. The imbalanced class distribution forces the learning algorithm to focus on majority classes and ignore the minority classes (Cristóbal Romero, Ventura, Espejo, et al., 2008). Thus, most of the time the classifiers leads to a wrong conclusion. In such situation, it would be more suitable to apply another measure such as, Geometric Mean, Lift, ROC, AUC, Precision, Specificity, Recall (sensitivity) and F-measure (Jovanovic et al., 2012; Kularbphettong & Tongsiri, 2014; Cristóbal Romero, Ventura, Espejo, et al., 2008; Zafra, Romero, & Ventura, 2011).

Generally, there is no standard metrics for evaluating the performance of classification algorithms. The choice of choosing a measurement method is still a subjective concept. Although, sometimes the nature of data or the type of algorithms involved can influence the decision of selecting a particular metric(s), typically it is up to the researcher to decide which measurements methods are suitable for his/her research..

Table 4-2 shows the most common qualitative measures used for classifiers evaluation. These measures are built by computing the values recorded on the confusion matrix (see Table 4-1).

Table 4-1: Confusion Matrix

		Actual (true class)	
		Positive	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Table 4-2: Performance metrics

Accuracy	$(TP + TN)/(TP + FP + FN + TN)$
Precision	$TP/(TP + FP)$
Recall (sensitivity)	$TP/(TP + FN)$
Specificity	$TN/(FP + TN)$
F-measure	$[(1 + \beta^2) \times recall \times precision]/[(\beta^2 \times recall) + precision]$
F-measure (F1-score)	$(2 \times recall \times precision)/(recall + precision)$
g-mean	$\sqrt{recall \times precion}$
AUC	$(Sensitivity + \ Specificity)/2$

Since this experiment is not dealing with imbalanced data, accuracy measure were used for assessing the quality of different classifiers. By definition, Accuracy measures the fraction of correctly classified instances (A. B. M. S. Ali & Wasimi, 2007).

4.2.1.1. Experimental set-up for classifiers performance evaluation

This experiment was conducted to evaluate the performance of different classification algorithms. The aim of this experiment was to build and compare different predicting models to classify the final grade (FAIL or PASS) based on the student's assessments (assignments and quizzes).

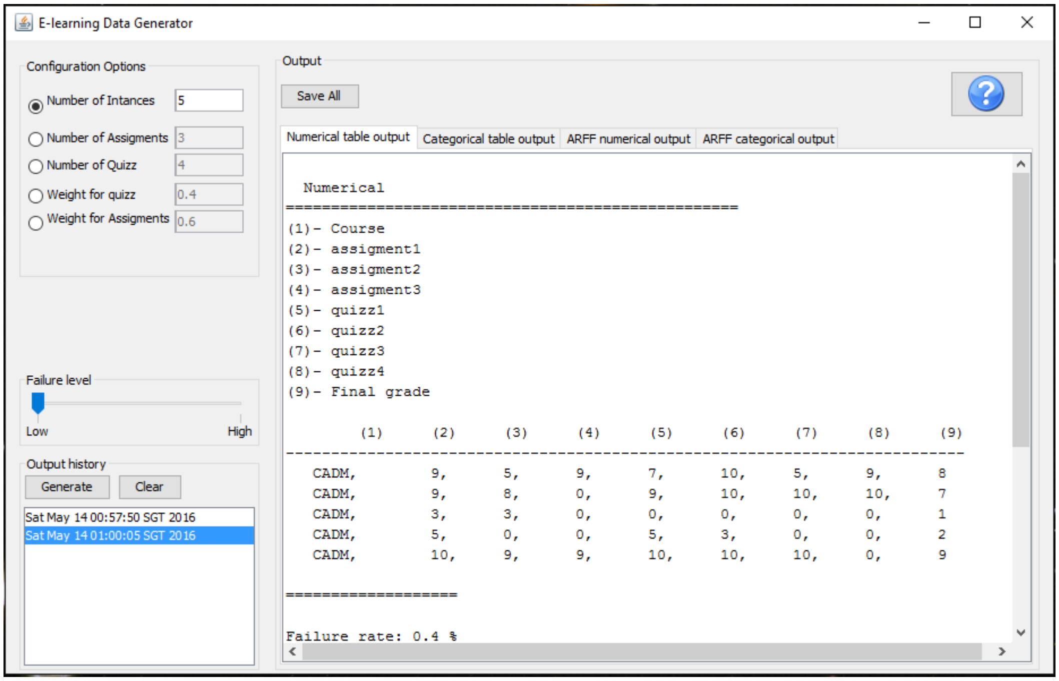
For the experiment, simulated data were used. The simulated data were generated through the EDG (E-learning Data Generator) application. The EDG is an application developed for generating E-learning simulated data (see Figure 4.1). This application randomly assigns values in the range between zero and ten for each student assessment (quizzes and assignments), and then computes the final grade based on the following formula:

$$grade = \left(\frac{\sum_{i=1}^{nQ} random(0,10)}{q} \times w_1\right) + \left(\frac{\sum_{i=0}^{nA} random(0,10)}{a} \times w_2\right) \qquad (1)$$

where, *nQ* is the number of quizzes taken by a student, *q* is the user defined quantity of quizzes needed for each student to sit for in specific course, *w1* is the user defined weight for quizzes, *nA* is the number of assignments taken by a student, *a* is the user defined quantity of assignments needed for each student to sit for in specific course, *w2* is the user defined weight for assignments, and *Random()* is a function that randomly assigns a value for assessment. The *Random()*function generates uniformly distributed random numbers for students' assessments, the function uses the following Monte Carlos formulation (Rubinstein & Kroese, 2008) for generating uniform values for student's assessments:

$$value = R * (b - a) + a \qquad (2)$$

where, *a=0* and *b=10* are the limits of the issue numbers, and *R* is a uniform random number. When the *value* =0, indicates that the student does not do this assessment.



● Number of Intances 5
○ Number of Assigments 3
○ Number of Quizz 4
○ Weight for quizz 0.4
○ Weight for Assigments 0.6

Sat May 14 00:57:50 SGT 2016
Sat May 14 01:00:05 SGT 2016

```
  Numerical
==================================================
(1)- Course
(2)- assigment1
(3)- assigment2
(4)- assigment3
(5)- quizz1
(6)- quizz2
(7)- quizz3
(8)- quizz4
(9)- Final grade

        (1)      (2)      (3)      (4)      (5)      (6)      (7)      (8)      (9)
---------------------------------------------------------------------------------------------
   CADM,        9,       5,       9,       7,       10,      5,       9,       8
   CADM,        9,       8,       0,       9,       10,      10,      10,      7
   CADM,        3,       3,       0,       0,       0,       0,       0,       1
   CADM,        5,       0,       0,       5,       3,       0,       0,       2
   CADM,        10,      9,       9,       10,      10,      10,      0,       9

===================

Failure rate: 0.4 %
```

Figure 4.1: E-learning Data Generator GUI

EDG tool was used to generate 500 instances with each one containing student's results of three assignments (60% worth), four quizzes (40% worth) and the final grade of a single course. The data was summarized and transformed into ARFF transactional file, which is the Weka's native data file for describing the list of instances sharing a set of attributes. As a result, two datasets (numerical and discrete) were created, both containing six attributes (see Table 4-3) along with a nominal class attribute (final grade) in ordinal labelling: "FAIL" (when the final grade outcome is less than five), and "PASS" (when the final grade outcome is greater or equal to five). The data distribution was balanced, with 57% instances FAIL and 43% instances PASS.

Table 4-3: List of dataset's attributes

Attribute name	Description
n_assign_taken	Number of assignment taken
n_assign_fail	Number of assignment failed
n_assig_pass	Number of assignment passed
n_quiz_taken	Number of quizzes taken
n_quiz_pass	Number of quizzes passed
n_quiz_fail	Number of quizzes failed
final_grade (class)	Final grade the student obtained in course.

The discrete dataset was created by discretizing the numerical data using Binning/Equal-width algorithm (Witten & Frank, 2005). The data was divided into three intervals: LOW, MEDIUM and HIGH. Weka Experimenter tool was used to test and compare different state-of-art classification algorithms performance by applying the default parameters and random seeds. Eleven algorithms were applied to handle with both numeric and discrete datasets, namely, ZeroR (Zero Rule), PART(Partial decision tree algorithm), Ridor (RIpple DOwn Rule Learner), SimpleCART (CART algorithm), J48 (C4.5 implementation in Weka), REPTree (Reduced Error Pruning Tree), Logistic (Regression), MultilayerPerceptron (i.e. NN – Neural Network), SMO (Sequential Minimal Optimization), BayesNet (Bayes Networks), NaiveBayes (Naïve Bayes).

Because the dataset was less than 1000 instances, 10-folds cross-validation technique was used to achieve an unbiased estimate model performance, and avoid the overfitting problem. This methods randomly divides the dataset into ten subsets, and runs 10 trials. In each iteration, one subset is used for testing and the other nine subsets are used for training (A. B. M. S. Ali & Wasimi, 2007). The average error across all 10 trials is then computed.

4.2.2. Classifiers comprehensibility evaluation

According to Kohavi (1996), Comprehensibility or interpretability, is the ability to understand the output of induction algorithm. The motivation for comprehensible classification models in education is because most of users usually encountered in educational context have few or almost no expertise in DM. Thus, a classifier model appropriate for an educational environment has to be both accurate and comprehensible in order for instructors and course administrators to be able to use it for decision making (Cristobal Romero et al., 2010). However, comprehensibility is a kind of subjective concept, since a classification model can be little comprehensible for one user but very comprehensible for another user (Pappa & Freitas, 2010). Therefore, in order to avoid difficult subjective issues, many literature in DM literature often uses an objective measure of "simplicity," based on classification model size "– *in general the smaller the classification model, the better the model. In particular, when the classification model is expressed by IF-THEN classification rules, rule set size is usually used as a proxy to comprehensibility: in general, the smaller a rule set – i.e., the smaller the number of rules and the number of conditions per rule – the simpler the rule set is deemed to be. This concept of rule comprehensibility is still popular in the literature, probably due to its simplicity and objectivity, since the system can easily count the number of rules and rule conditions without depending on a subjective evaluation of a rule or rule set by the user. However, rule length is clearly a far from perfect criterion for measuring rule comprehensibility*" (Pappa & Freitas, 2010).

Subjectivity, on the other hand, appears to be the best way to evaluate the classifiers comprehensibility (Martens & Baesens, 2010). Usually, a subjective evaluation is done by asking the application domain experts and users for their own ranking to determine their "mental fit" with the different output of induction algorithm (Martens & Baesens, 2010). Therefore, for the comprehensibility evaluation experiment, this study designed a comprehensibility survey in order to get the user's subjective opinion regarding the comprehensibility of the classification models.

4.2.2.1. Survey design for comprehensibility evaluation

All rule-based and tree-based models built from the classifiers performance evaluation experiment were transformed into IF-THEN rules knowledge representation. Then, a survey to measure subjectively the comprehensibility of each rule was created and distributed among different respondents asking them to give their subjective opinion about the rules on a scale with five levels, namely:

- Very easy to comprehend;
- Easy to comprehend;
- Comprehensible;
- Difficult to comprehend;
- Very difficult to comprehend.

In total 56 rules were created, each rule matching with the following pattern:

IF *n_assign_taken | n_assign_fail | n_assign_pass | n_quiz_taken | n_quiz_fail | n_quiz_pass*
AND *...*
THEN *final_grade*

Where *assign, n_assign_fail, n_assign_pass, n_quiz, n_quiz_fail, nquiz_pass* and *final_grade* are thereby generic attributes referring to:

- *n_assign_taken* – Number of assignment taken (HIGH, MEDIUM and LOW values and numeric values);

- *n_assign_fail* – Number of assignment failed (HIGH, MEDIUM and LOW values or numeric values);
- *n_assign_pass* – Number of assignment passed (HIGH, MEDIUM and LOW values or numeric values);
- *n_quiz_taken* – Number of quizzes taken (HIGH, MEDIUM and LOW values or numeric values);
- *n_quiz_fail* – Number of quizzes failed (HIGH, MEDIUM and LOW values or numeric values);
- *n_quiz_pass* – Number of quizzes passed (HIGH, MEDIUM and LOW values or numeric values);
- and lastly, *final_grade* refers to the final grade the student obtained in course (FAIL and PASS values).

The completed questionnaires were received from 12 respondents out of a total of 13 respondent contacted.

4.3. Performance analysis

The second experiment was to evaluate the overhead introduced by the SOA mechanisms and the distributed knowledge discovery with respect to the overall execution time. The second experiment seeks to understand the SOA mechanism and performance properties of proposed framework by finding the answer to the following research questions:

- What are the execution times of different steps needed to perform a typical EDM task in different LAN (Local Area Network) scenarios?
- What is the efficiency of the distributed knowledge discovery mechanism to automate and simplify the development of EDM in E-learning environments?

4.3.1. Experimental set-up

The experiment were performed using two machines (see Figure 4.2):

- The client. With 1.60 GHz processor INTEL (R) core (TM) and 4GBytes of RAM, running Linux Ubuntu 14.04.
- The server (providing Web services). With 1.80 GHz processor INTEL (R) core (TM) and 4GBytes of RAM, running Windows 10.

The experiment took place in two Local Area Network (LAN) scenario, as follows:

- Scenario 1: Average bandwidth of 31 Mbps.
- Scenario 2: Average bandwidth of 4 Mbps.

The execution times of different steps needed to perform a typical EDM task were measured in order to learn something from the system behaviour. More precisely, the execution of the following steps, (see Figure 4.2):

1. Data gathering and transformation – The SOEDM invokes the *local_miningdata_get_students_data* service from MDPWS to collect data from a Moodle connection. The *local_miningdata_get_students_data* service responds with an XRAD dataset. Then, the SOEDM invokes the *xmlToArrf* service from WDMWS to transform the XRAD dataset into ARFF format. The *xmlToArrf* service responds with a new dataset in ARFF notation (see Table 3-2 and Table 3-3).
2. Data Filtering – The SOEDM invokes the *executeFilter* operation from WDMWS to filter the data. The operation receives as parameters the data (in ARFF format),

and the filtering scheme and its options (see Table 3-4). The *executeFilter* service responds with with a new dataset in ARFF notation (see Table 3-5).

3. Classification – The SOEDM invokes the *executeClassifier* operation from WDMWS to execute a classification task. The operation receives as parameters the data (in ARFF format), the classification scheme and its options, and evaluation options (see Table 3-6). The *executeClassifier* service responds with the output of the remote execution of classification task (see Table 3-7).
4. Clustering – The SOEDM invokes the *executeCluster* operation from WDMWS to execute a cluster task. The operation receives as parameters the data (in ARFF format), the cluster scheme and its options, and evaluation options (see Table 3-8). The *executeCluster* service responds with the output of the remote execution of cluster task (see Table 3-9).
5. Association – The SOEDM invokes the *executeAssociate* operation from WDMWS to execute an association mining task. The operation receives as parameters the data (in ARFF format), the association scheme and its options (see Table 3-10). The *executeAssociate* service responds with the output of the remote execution of cluster task (see Table 3-11).

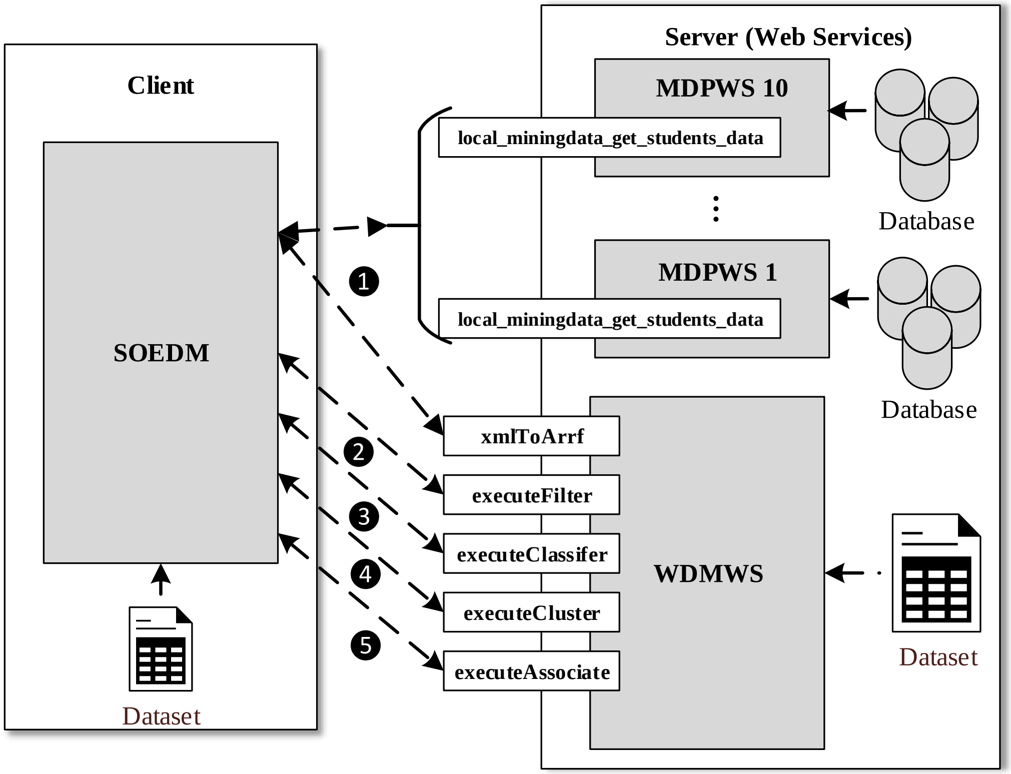


Figure 4.2: Execution of EDM tasks on EDM-framework

EDG tool was used to generate 5 data sets of different size. Each data set containing student's results of three assignments (60% worth), four quizzes (40% worth) and the final grade of a single course (see Table 3-1). Then, each of these data set (in .CSV format) was introduced into different Moodle systems as if they were the results of a particular course, thereby, 5 different Moodle learning environment were created.

Table 4-4 shows the characteristics of the 5 generated data sets, considering that zero to ten (0-10) as the assessment grading scale range, and six (6) as the minimum passing score.

Table 4-4: Characteristics of the generated datasets

	Number of Instances	Number of students that passed the course	Number of students that failed the course
1	1000	440	560
2	2000	837	1163
3	3000	1879	1121
4	4000	1670	2330
5	5000	2109	2891

The SOEDM client was used for connecting each one of the 5 Moodle environments. Once the connection was established, the SOEDM was used to execute each of the five steps mentioned above (i.e., data gathering and transformation, data filtering, classification, clustering, and association), where, *Remove* (passing *–R 1-3* parameter) and *ManualDiscretize* (passing *-A low medium high -C fail pass* parameters) filtering algorithms were used for filtering the data; *J48* algorithm (passing default parameters) was used for classification analysis; *SimpleKMeans* algorithm (passing default parameters) was used for clusters analysis; and, *Apriori* algorithm (passing default parameters) was used for association analysis.

Each execution of the above steps was computed the average execution times from 20 runs. The outcome of the experiment are the average execution times of the 5 EDM task.

4.4. Summary

This chapter has provided the experimental design which is subject to the research questions of this study. Two experiments were designed in order to respond the research questions. In addition, the experimental set up for both experiments is explained.

The first experiment aims to guarantee that the students' performance prediction model is based upon an accurate model, also ensure that the model is friendly to educators so that they are able to interpret and understand the model.

The second experiment aims to evaluate the overhead introduced by the SOA mechanisms and the distributed knowledge discovery with respect to the overall execution time.

CHAPTER 5

5. Results and Discussion

5.1. Introduction

This chapter is divided into two sections. The first section reports the experimental results of the proposed method in predicting student's final marks. The performance of different prediction models is evaluated by using two sets of student’s usage data (numeric and discrete values). Where, the performances of the models are evaluated based on the Accuracy criteria. Then, the higher ranked algorithms are measured subjectively in terms of their comprehensibility by means of a survey where different respondents provided their subjective opinion about the model's comprehensibility on a scale with five levels (Very easy to comprehend, Easy to comprehend, Comprehensible, Difficult to comprehend, and Very difficult to comprehend).

The second section evaluate the overhead introduced by the SOA mechanisms and the distributed knowledge discovery with respect to the overall execution time.

5.2. Classifiers performance and comprehensibility evaluation

The measure reported in Table 5-1 refer to the performance evaluation of different prediction models to classify the final grade based on two sets of student’s usage data. It is noted that noted that this experiment is primary focused on rule-based and tree-based models. Tree-based and Rule-based models are rather comprehensible than black box models, such as Function-based or Bayes-based models (Dole & Jayant, 2014; García, Romero, Ventura, & Castro, 2007; Cristobal Romero et al., 2010). However, function-based and bayes-based models has been included in the performance evaluation in order to compare their relative performance over the rule-based and tree-based models.

The experimental results showed that models built from numerical dataset performed slightly better than models built from the discrete dataset. Among the algorithms, tree-based and rule-based algorithms showed better performance than other models, overall in which they achieved a higher ranking in accuracy, with the exception of the baseline algorithm ZeroR that reached only 57% in accuracy.

Table 5-1: Performance results

Algorithm		Accuracy	
		Discrete	Numeric
Rule	ZeroR	0.5677	0.5677
	Ridor	0.8839	0.93
	PART	0.8892	0.9329
Trees	CART	0.8946	0.9303
	J48	0.8932	0.9325
	REPTree	0.891	0.9321
functions	Logistic	0.893	0.9342
	NN	0.887	0.9307
	SMO	0.8884	0.9325
bayes	BayesNet	0.8705	0.8603
	NaiveBayes	0.8705	0.9088

The findings for the first part of the experiment is that on the basis of accuracy results, the selected models are: PART (Disc=89%, Num=93%), Ridor (88%, Num=93%), CART (89%, Num=93%), J48 (89%, Num=93%), and REPTree (89%, Num=93%). These models were transformed into a set of IF-THEN rules and distributed among different respondents for rating the rules comprehensibility, then resulting in the graph depicted in Figure 5.1.

As shown in Figure 5.1, it is clear that the PART model was the most ranked model with respect to comprehensibility and predicting accuracy. Since, PART obtained a good prediction accuracy (Disc=89%, Num=93%) and approximately 66% of the respondents rated the PART model’s comprehensibility at or above an “easy to comprehend” level. Thus, this study suggested that it would be beneficial if the PART model can be used to predict student's final marks based on the students' usage data.

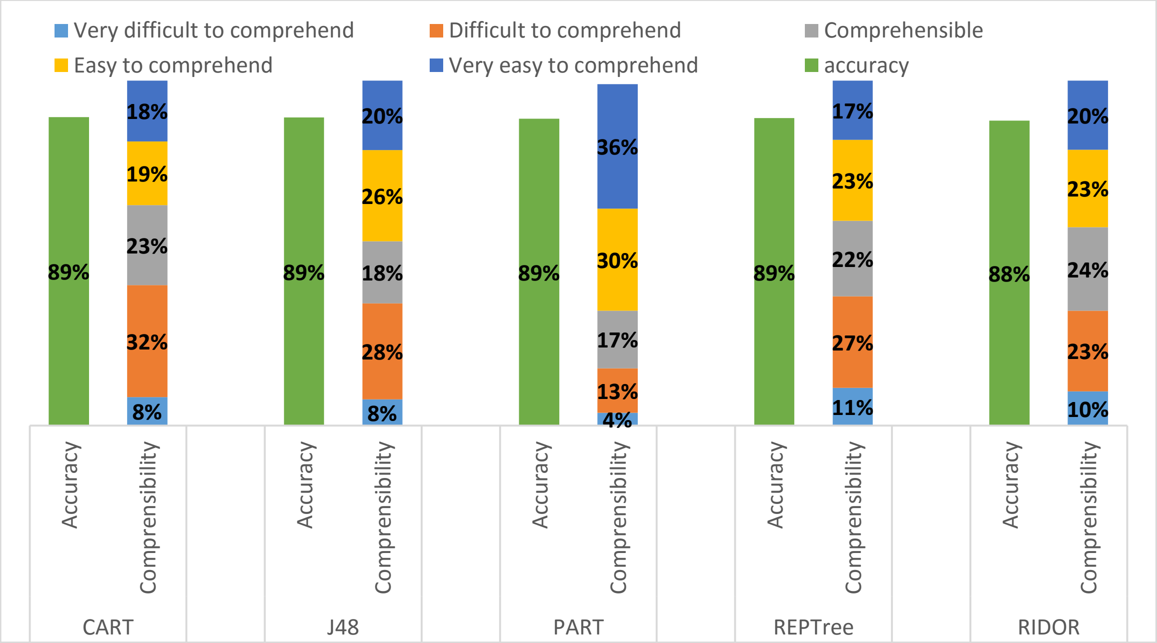


Figure 5.1: Subjective opinion about comprehensibility of rule-based and tree-based classifiers

5.3. Performance analysis

The performance analysis experiment was executed both in the scenario 1 (with an average bandwidth of 31 Mbps) and scenarios 2 (with an average bandwidth of 4 Mbps). Five datasets were used, containing a number of instances ranging from 1000 to 5000. For each dataset size and network scenario, 20 independent executions were run. The measures reported in the following tables and graphs resulted from the average values of all executions.

Table 5-2 and Figure 5.2 shows the execution times of the different steps in the LAN scenario 1 (with average bandwidth of 31 Mbps) for datasets sizes ranging from 1000 to 5000 instances, whereas Table 5-3 and Figure 5.3 shows the execution times of the different steps in the LAN scenario 2 (with average bandwidth of 4 Mbps) for datasets sizes from 1000 to 5000 instances.

Table 5-2: Execution times (in milliseconds) needed to complete the different phases

Scenario 1 (with average bandwidth of 31 Mbps)					
	Average execution times (in millisecond)				
Number of instances	1000	2000	3000	4000	5000
Data gathering and transformation	29531.2	60541.85	91504.05	120383.15	153784.1
Filtering	404.4	851.45	1194.4	1881.45	2294.95
Classification	87.5	183.2	220.85	322.9	472.15
Clustering	58.15	71.8	69.75	86.85	112.15
Association	97.85	157.85	244.2	303.85	431.45

Table 5-3: Execution times (in milliseconds) needed to complete the different phases

Scenario 2 (with average bandwidth of 4 Mbps)					
	Average execution times (in millisecond)				
Number of instances	1000	2000	3000	4000	5000
Data gathering and transformation	36801.5	87554.5	148816.4	237196.3	289081.65
Filtering	698.2	2118.55	4604.8	8353.45	15076.35
Classification	171.25	491.05	1781.7	4187.55	6679.9
Clustering	61.5	73.9	78.75	101.1	104.95
Association	153.15	853.6	1448.65	5416.2	10274.15

In both tables (Table 5-2 and Table 5-3) and graphs (Figure 5.2 and Figure 5.3), it is relatively clear that the execution time of the *data gathering and transformation* task increase with the growth of the dataset size. The similar result is shown in Figure 5.2 and Figure 5.3 where the execution times of the four tasks – *Filtering, Classification, Clustering, and Association* – increases with the growth of the dataset size.

Besides, it is also shown in Table 5-2 and Table 5-3 that the higher is the network bandwidth speed, the faster is the execution times for performing EDM tasks. For instance, while in Table 5-2 took in average 29531.2 milliseconds to execute the data gathering and transformation tasks in a dataset with 1000 instances, in Table 5-3 (less bandwidth) the time used to execute the same tasks were approximately 36801.5 milliseconds, which is relatively slower. This pattern repeats throughout the other tasks and different dataset sizes.

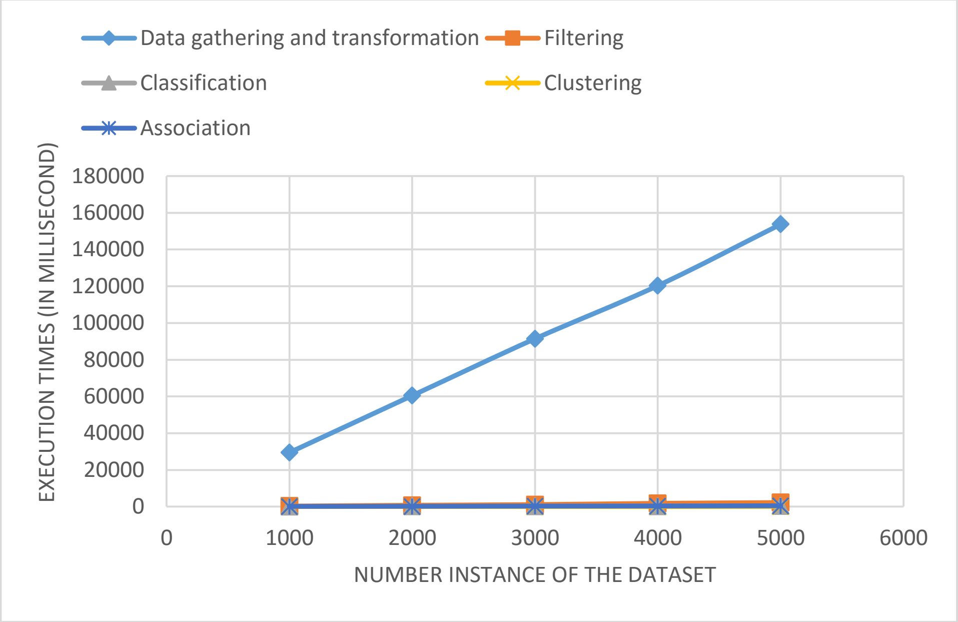


Figure 5.2: Execution times of the 5 EDM tasks in the scenario 1

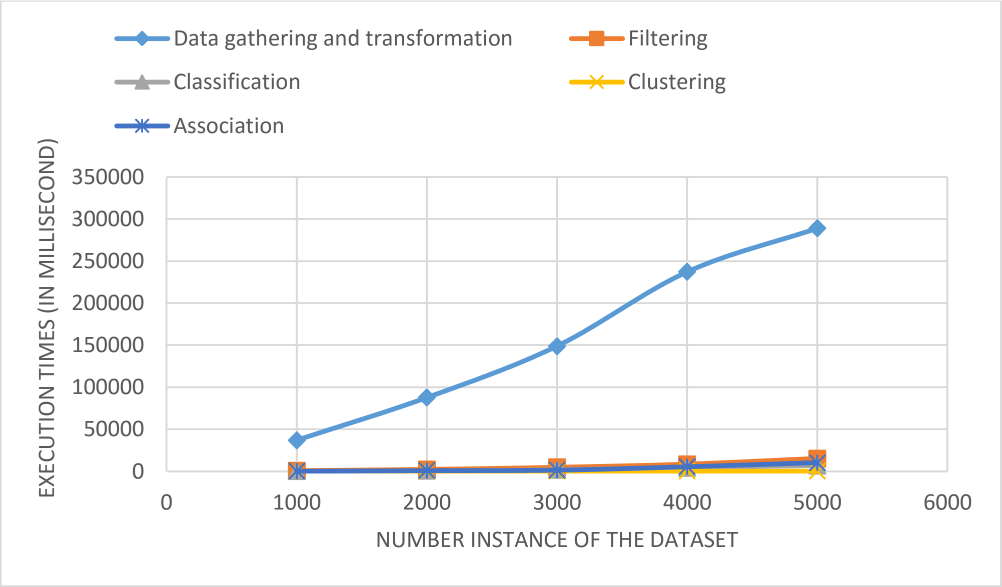


Figure 5.3: Execution times of the 5 EDM tasks in the scenario 2

Figure 5.4 shows the similar result as in Figure 5.2, except that the “data gathering and transformation” tasks were not shown for clear visualization purpose.

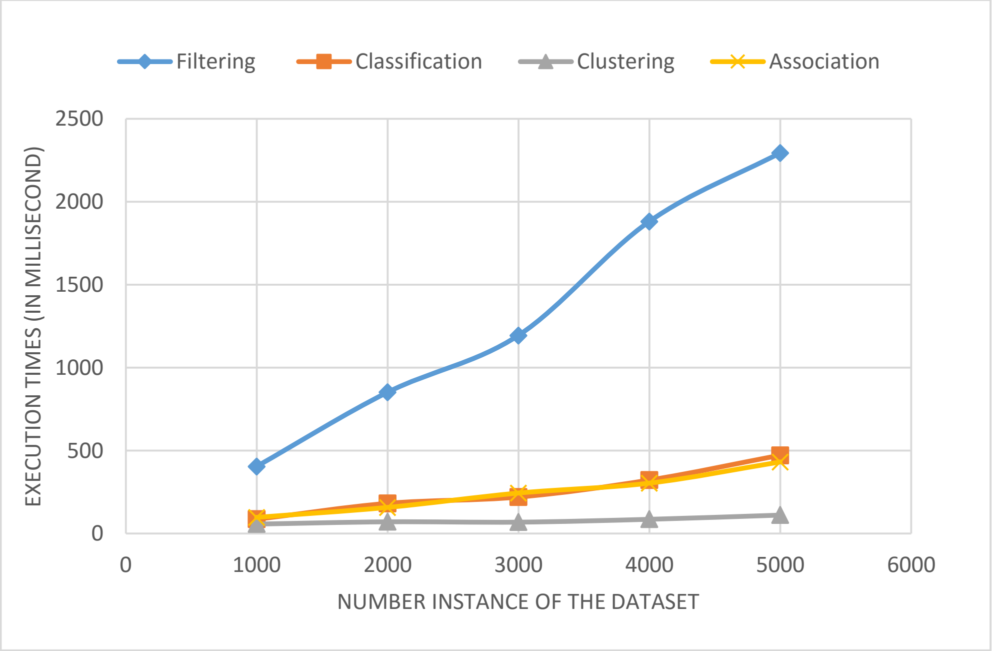


Figure 5.4: Execution times of the 4 EDM tasks in the scenario 1

Figure 5.5 shows the similar result as in Figure 5.3, except that the “data gathering and transformation” tasks were not shown for clear visualization purpose.

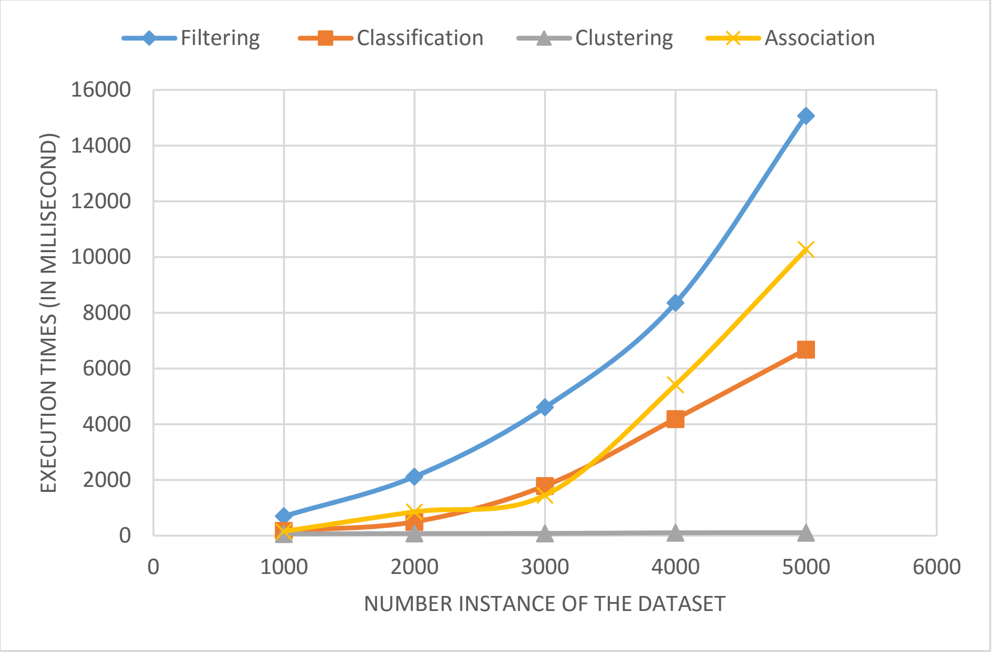


Figure 5.5: Execution times of the 4 EDM tasks in the scenario 2.

To better highlight the overhead introduced by the SOA mechanisms and the distributed scenario, Figures 5.6 and 5.7 show the percentage of *data gathering and transformation*, and *operations overhead* (i.e., the sum of *Filtering, Classification, Clustering, and Association* tasks) with respect to the total execution time in the scenarios 1 and 2. In the scenario 1, that the *data gathering and transformation* takes averagely 98% of the total execution times, while the *operations overhead* takes averagely 2%. In the scenario 2, on the other hand, the execution times of *data gathering and transformation* ranges from 97% to 90% as the dataset size grow, while the *operations overhead* ranges from 3% to 10%. In general, the overhead of the data mining operations – *Filtering, Classification, Clustering, and Association* – is relatively small, especially in scenarios with high bandwidth speed, or dealing with small datasets for which the *data gathering and transformation* is fast. Thus, it can be concluded that SOA mechanism can be effectively used to develop services for EDM applications.

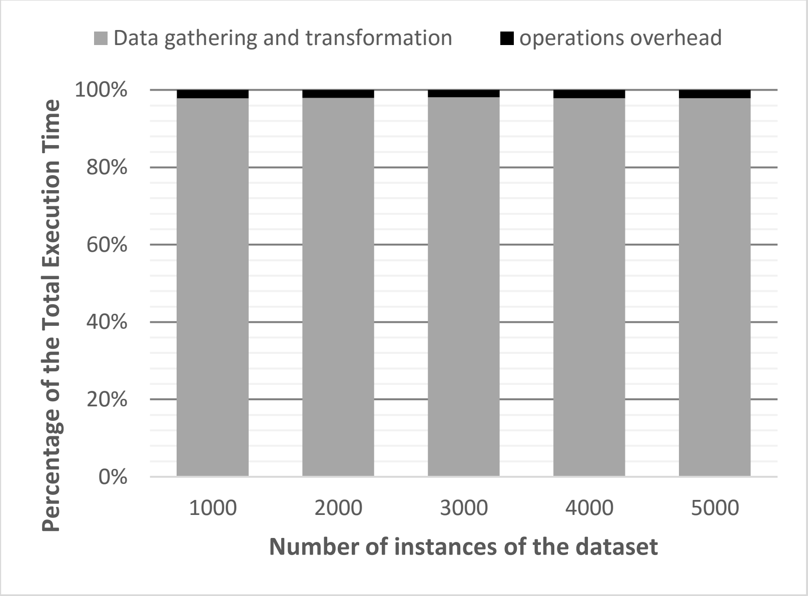


Figure 5.6: Percentages of the total execution time of the different steps in the scenario 1

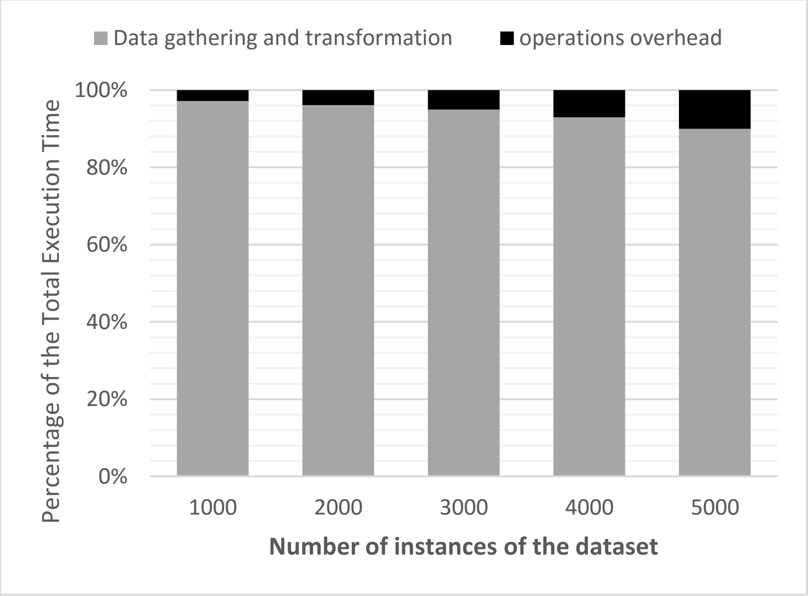


Figure 5.7: Percentages of the total execution time of the different steps in the scenario 2

5.4. Summary

In this chapter, the results obtained from the experimental work conducted are discussed. The chapter starts by examining the performance and comprehensibility results of different classification. Then, it is examined the overhead introduced by the SOA mechanisms and the distributed knowledge discovery with respect to the overall execution time.

The discussion is based on two experiments. In the first experiment, the results shown that PART outperforms the other models with the best accuracy and comprehensibility. On the other hand, observation of the second experiment demonstrated that the execution time is influenced by the size of the dataset, or by the network bandwidth speed: the bigger is the dataset size, the higher is the execution times; the higher is the network bandwidth speed, the faster is the execution times. Also, the results shown the overhead introduced by the SOA mechanism is relatively small, thus, it has been concluded that a SOA can be effectively used to facilitate EDM on E-learning environments.

CHAPTER 6

6. Conclusions and Future Work

6.1. Conclusion

This thesis addresses the issues surrounding the slow adoption of Educational Data Mining on E-learning settings due to the complexity around the currently existing EDM tools. The goal of this research was to design and implement an EDM application framework which aims at simplifying and automate the development of EDM in E-learning environments. With this goal in mind, this study proposed the use of distributed knowledge discovery as a mechanism to hide complexity underlying the EDM technical details, thus, enabling users to perform EDM in an automated fashion.

Based on the review of the literature and in-depth analysis, this study compared different state-of-art distributed knowledge discovery approaches and came to the conclusion that the Service-Oriented Architecture (SOA) is the most suitable solution for implementation of a distributed knowledge discovery EDM tool. Therefore, it was proposed an SOA-based EDM application framework, named EDM-framework that relies on abstraction, extensibility, and interoperability principles. Hence, providing the basis for the implementation of a reusable, extensible, maintainable, and above all an easy-to-use EDM tool. In order to demonstrate the feasibility of the proposed framework, the first prototype was built. It consists in modular architecture that allows to dynamically locate E-learning data provider services, and remotely run knowledge discovery tasks on data mining services. Also, allows saving the outputs models in a separate file in the form of rules that can be manually uploaded into a Moodle knowledge base. On the basis of these models stored in Moodle knowledge base, educators in Moodle system can predict their students' performances without the need to have technical skills or expertise in data mining.

The result shows that the overhead introduced by the SOA mechanism is relatively small, therefore, it has been concluded that a service-oriented architecture can be effectively used to facilitate educational data mining on E-learning environments.

6.2. Future Works

There are several promising directions to extend the work presented in this thesis:

1. Extend the current prototype to include more libraries and allow connections to more E-learning systems, and data mining engines.
2. Contribute the proposed data template Web service to the Moodle community so that the service is included in the Moodle Core Service of the next releases of Moodle.
3. Build an inference engine and knowledge base in Moodle system that uses rule-based output models to facilitate knowledge discovery in an intuitive manner for non-expert users in data mining such as educators, students and administrators.
4. Make the prototype freely available to public domain, so that future developers or EDM researchers can contribute to the toolbox.

REFERENCES

Albashiri, K. A., & Coenen, F. (2009). The EMADS extendible multi-agent data mining framework. *Data Mining and Multi-Agent Integration*, 189–200. http://doi.org/10.1007/978-1-4419-0522-2_13

Ali, A. B. M. S., & Wasimi, S. A. (2007). *Data Mining: Methods and Techniques*. South Melbourne, Australia: Thomson.

Ali, A. S., Rana, O. F., & Taylor, I. J. (2005). Web Services Composition for Distributed Data Mining. *2005 International Conference on Parallel Processing Workshops (ICPPW'05)*, 11–18. http://doi.org/10.1109/ICPPW.2005.87

Ali, F. M. D. A., & Ng, S. C. (2015). Moodle Data Retrieval for Educational Data Mining. *International Journal of Scientific Engineering and Technology*, *11*(4), 523–525.

Altorf, M. (2007). *Data mining on grids*. Universiteit Leiden.

Angoss. (2015). Retrieved from http://www.angoss.com

Bakharia, A., & Dawson, S. (2011). SNAPP: A Bird's-Eye View of Temporal Participant Interaction AneeshaBakharia. *Proceedings of the 1st International Conference on Learning Analytics and Knowledge - LAK '11*, 168. http://doi.org/10.1145/2090116.2090144

Barahate, S. R. (2012). Educational Data Mining as a Trend of Data Mining in Educational System. In *Proceedings of IJCA International Conference and Workshop on Emerging Trends in Technology* (pp. 11–16).

Birant, D. (2011). Service-Oriented Data Mining. In P. K. Funatsu (Ed.), *New Fundamental Technologies in Data Mining* (InTech). http://doi.org/10.5772/14066

Booch, G., Rumbaugh, J., Jacobson, I., & Matter, F. (1998). *The unified modeling language user guide*. *[the ultimate tutorial to the UML from the original designers]*. Addison-Wesley. Retrieved from Addison Wesley - Booch et al. - The Unified Modeling Language User Guide.pdf

Bouchet, F., Azevedo, R., Kinnebrew, J. S., & Biswas, G. (2012). Identifying Students' Characteristic Learning Behaviors in an Intelligent Tutoring System Fostering Self-Regulated Learning. In K. Yacef, O. R. Zaïane, A. Hershkovitz, M. Yudelson, & J.

Stamper (Eds.), *Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012)* (pp. 65–72). Chania, Greece.

Bousbia, N., & Belamri, I. (2014). Which Contribution Does EDM Provide to Computer-Based Learning Environments : Educational Data Mining, *524*, 3–28. http://doi.org/10.1007/978-3-319-02738-8

Bousbia, N., Rebaï, I., Labat, J. M., & Balla, A. (2010). Learners' navigation behavior identification based on trace analysis. *User Modeling and User-Adapted Interaction*, *20*(5), 455–494. http://doi.org/10.1007/s11257-010-9081-5

Braz, C. (2005). THE ROLE OF GRID COMPUTING IN E-LEARNING ". In *PROJECT I – "RGridE-LEarning: The Role of Grid Computing in E-Learning"* (pp. 1–36).

Brezany, P., & Hofer, J. (2003). Gridminer: An infrastructure for data mining on computational grids. In *the APAC Conference and Exhibition on Advanced Computing, Grid Applications and eResearch* (pp. 1–19). Queensland Australia.

Brezany, P., Janciak, I., & Tjoa, a. M. (2005). GridMiner: A fundamental infrastructure for building intelligent grid systems. In *Proceedings - 2005 IEEE/WIC/ACM InternationalConference on Web Intelligence, WI 2005* (Vol. 2005, pp. 150–156). http://doi.org/10.1109/WI.2005.68

Calders, T., & Pechenizkiy, M. (2012). Introduction to the special section on educational data mining. *ACM SIGKDD Explorations Newsletter*, *13*(2), 3. http://doi.org/10.1145/2207243.2207245

Cannataro, M., & Talia, D. (2003). The knowledge grid. *Association for Computing Machinery. Communications of the ACM*, *46*(1), 89. http://doi.org/10.1145/602421.602425

Cesario, E., & Talia, D. (2008). Distributed data mining models as services on the grid. *Proceedings - IEEE International Conference on Data Mining Workshops, ICDM Workshops 2008*, 486–495. http://doi.org/10.1109/ICDMW.2008.29

Chen, X., Williams, G., & Xu, X. (2007). A Survey of Open Source Data Mining Systems. *Emerging Technologies in Knowledge Discovery and Data Mining*, *4819*(60603066), 3–14. http://doi.org/10.1007/978-3-540-77018-3

Chen, Z., Liui, S., & Liu, G. (2008). The Multi-Agent Knowledge Management System Model for Pervasive Computing. In *3rd international conference on pervasive computing and applica- tions* (pp. 70–73). Alexandria, Egypt.

Cheung, W. K., Zhang, X., Luo, Z., & Tong, F. C. H. (2006). Service-Oriented Distributed Data Mining. *Ieee Internet Computing*, (August), 44–54.

Cocea, M., Hershkovitz, A., & Baker, R. S. J. D. (2009). The impact of off-task and gaming behaviors on learning: Immediate or aggregate. In *Frontiers in Artificial Intelligence and Applications* (Vol. 200, pp. 507–514). http://doi.org/10.3233/978-1-60750-028-5-507

Corbett, a, & Anderson, J. (1995). Knowledge-tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User Adopted Interaction*, *4*, 253–278.

Dam, H. H., Abbass, H. A., & Lokan, C. (2005). DXCS: an XCS System For Distributed Data Mining. In *Proceedings of the 2005 conference on Genetic and evolutionary computation - GECCO '05* (p. 1883). http://doi.org/10.1145/1068009.1068326

DeFreitas, K., & Bernard, M. (2014). A Framework for Flexible Educational Data Mining. In *Proceedings of the International Conference on Data Mining (DMIN)* (p. 1).

Desmarais, M. (2012). Mapping Question Items to Skills With Non-Negative Matrix Factorization. *ACM SIGKDD Explorations Newsletter*, *13*(2), 30–36. http://doi.org/10.1145/2207243.2207248

Dole, L., & Jayant, R. (2014). A Decision Support System for Predicting Student Performance. *International Journal of Innovative Research in Computer and Communication Engineering*, 7232–7237.

Doran, P. R., Doran, C., & Mazur, A. (2011). Social network analysis as a method for analyzing interaction in collaborative online learning environments. *Journal of Systemics, Cybernetics and Informatics*, *9*(7), 10–16.

Dringus, L. P., & Ellis, T. (2005). Using data mining as a strategy for assessing asynchronous discussion forums. *Computers and Education*, *45*(1), 141–160. http://doi.org/10.1016/j.compedu.2004.05.003

Dyckhoff, A. L., Zielke, D., Bültmann, M., Chatti, M. A., & Schroeder, U. (2012). Design and implementation of a learning analytics toolkit for teachers. *Educational Technology and Society*, *15*(3), 58–76.

García, E., Romero, C., Ventura, S., & Castro, C. De. (2007). An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modeling and User-Adapted Interaction*, *19*(1-2), 99–132.

Graf, S., Ives, C., Rahman, N., & Ferri, A. (2011). AAT: a tool for accessing and analysing students' behaviour data in learning systems. In *1st International Conference on Learning Analytics and Knowledge* (pp. 174–179). New York: ACM. http://doi.org/10.1145/2090116.2090145

Guedes, D., Meira, W., & Ferreira, R. (2006). Anteater: A service-oriented architecture for high-performance data mining. *IEEE Internet Computing*, *10*(4), 36–43. http://doi.org/10.1109/MIC.2006.69

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, *11*(1). Retrieved from http://www.cs.waikato.ac.nz/ml/weka/

He, W. (2013). Examining students' online interaction in a live video streaming environment using data mining and text mining. *Computers in Human Behavior*, *29*(1), 90–102. http://doi.org/10.1016/j.chb.2012.07.020

HEEPF. (2016). E-learning & Simulation Labs. Retrieved April 25, 2016, from http://www.heepf.org.eg/pdf/Booklets/E-learning & Simulation Labs.pdf

International Educational Data Mining Society. (2016). Retrieved July 3, 2016, from http://www.educationaldatamining.org/

Jeong, H., & Biswas, G. (2008). Mining Student Behavior Models in Learning-by-Teaching Environments. In *1st International Conference on Educational Data Mining* (pp. 127–136).

Johnson, M., & Barnes, T. (2010). EDM visualization tool: Watching students learn. *Proceedings of the 3rd International Conference on Educational Data Mining*, 297–298.

Jovanovic, M., Vukicevic, M., Milovanovic, M., & Minovic, M. (2012). Using data mining on student behavior and cognitive style data for improving e-learning systems : a case study. *International Journal of Computational Intelligence Systems*, *5*(3), 597–610.

Khan, D. (2008). CAKE - Classifying, Associating and Knowledge DiscovEry an approach for Distributed Data Mining (DDM) using PArallel Data Mining Agents (PADMAs). In *Proceedings - 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT Workshops 2008* (pp. 596–601). http://doi.org/10.1109/WIIAT.2008.236

Kleftodimos, A., & Evangelidis, G. (2013). An overview of Web Mining in Education. In *Proceedings of the 17th Panhellenic Conference on Informatics* (pp. 106–113). ACM.

Knime. (2015). Retrieved from http://www.knime.org/

Koh, H. C., & Tan, G. (2011). Data mining applications in healthcare. *Journal of Healthcare Information Management, 19*(2), 65.

Kohavi, R. (1996). Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid. In *Proceedings of the 2nd Int. Conf. on KD and DM* (pp. 202–207).

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *IEEE Computer Society Press*, 30–37.

Kularbphettong, K., & Tongsiri, C. (2014). Mining Educational Data to Support Students ' Major Selection, (1), 21–23.

Kumar, A., Kantardzic, M., Ramaswamy, P., & Sadeghian, P. (2004). An Extensible Service Oriented Distributed Data Mining Framework. In *International Conference on Machine Learning and Applications* (pp. 256 – 263). Louisville, Kentucky, USA: IEEE. http://doi.org/10.1109/ICMLA.2004.1383522

Lauría, E., & Baron, J. (2011). Mining Sakai to Measure Student Performance: Opportunities and Challenges in Academic Analytics. In *Proceedings of Enterprise Computing Community Conference* (pp. 1–16). Retrieved from http://ecc.marist.edu/conf2011/materials/LauriaECC2011- Mining Sakai to Measure Student Performance - final.pdf

Liu, B., Cao, S. G., & He, W. (2011). Distributed data mining for e-business. *Information Technology and Management*, *12*(2), 67–79. http://doi.org/10.1007/s10799-011-0091-8

López, M. I., Luna, J. M., Romero, C., & Ventura, S. (2012). Classification via clustering for predicting final marks based on student participation in forums. In *5th International Conference on Educational Data Mining, EDM 2012* (pp. 148–151). Chania, Greece.

Luo, J., Wang, M., Hu, J., & Shi, Z. (2007). Distributed data mining on Agent Grid: Issues, platform and development toolkit. *Future Generation Computer Systems, 23*(1), 61–68. http://doi.org/10.1016/j.future.2006.04.015

Luo, P., He, Q., Huang, R., & Lin, F. (2005). Execution Engine of Meta-learning System for KDD in Multi-agent Environment. In *Proceedings of the international workshop on autonomous intel- ligent systems: agents and data mining,* (pp. 149–160). St. Petersburg, Russia. http://doi.org/10.1007/11492870_12

Martens, D., & Baesens, B. (2010). Building Acceptable Classification Models. In R. Stahlbock, S. F. Crone, & S. Lessmann (Eds.), *Data Mining* (pp. 112–118). http://doi.org/http://dx.doi.org/10.1016/B978-0-08-044894-7.01318-X

Mazza, R., & Dimitrova, V. (2004). Visualising Student Tracking Data to Support Instructors in Web-Based Distance Education. In *International World Wide Web conference* (pp. 154–161). New York, USA.

Moodle. (2015). Retrieved from https://moodle.org

Moradi, H., Moradi, S. A., & Kashani, L. (2014). Students' Performance Prediction Using Multi-Channel Decision Fusion. In A. Pena (Ed.), *Educational Data Mining* (pp. 151–174). Springer International Publishing.

Mostow, J., & Beck, J. (2006). Some useful tactics to modify, map and mine data from intelligent tutors. *Natural Language Engineering*, *12*(2), 195. http://doi.org/10.1017/S1351324906004153

Mostow, J., Beck, J., Cen, H., Cuneo, A., Gouvea, E., & Heiner, C. (2005). An Educational Data Mining Tool to Browse Tutor-Student Interactions : Time Will Tell ! In *Proceedings of the Workshop on Educational Data Mining (2005)* (pp. 15–22). Pittsburgh, USA.

Nagappan, R., Skoczylas, R., & P. Sriganesh, R. (2003). *Developing Java Web Services*. Indianapolis, Indiana: Wiley.

Newcomer, E., & Lomow, G. (2005). Introduction to SOA with Web services. *Understanding SOA with Web Services*, 1–50.

Ontario, Eh. (2011). *Ontario' s eHealth Blueprint - in-depth*. Toronto, Ontario. Retrieved from www.ehealthblueprint.com

Orange. (2015). Retrieved from http://orange.biolab.si

Pappa, G. L., & Freitas, A. (2010). Automating the Design of Data Mining Algorithms. In *Natural Computing* (pp. 17–46). http://doi.org/10.1007/978-3-642-02541-9 2

Perez, M. S., Sanchez, A., Herrero, P., Robles, V., & Pena, J. M. (2005). Adapting the weka data mining toolkit to a grid based environment. In *Advances in Web Intelligence, Proceedings* (Vol. 3528, pp. 492–497). http://doi.org/10.1007/11495772

Rabbany, R., Takaffoli, M., & Za, O. R. (2011). Analyzing Participation of Students in Online Courses Using Social Network Analysis Techniques. In *4th International Conference on Educational Data Mining*.

RapidMiner. (2015). Retrieved from https://rapidminer.com/

Romero, C., Espejo, P. G., Zafra, A., Romero, J. R., & Ventura, S. (2010). Web usage mining for predicting final marks of students that use Moodle courses. *Computer Applications in Engineering Education*, *21*(1), 135–146. http://doi.org/10.1002/cae.20456

Romero, C., Romero, J. R., Luna, J. M., & Ventura, S. (2010). Mining Rare Association Rules from e-Learning Data. In *Proceedings of 3rd International Conference on Educational Data Mining* (pp. 171–180).

Romero, C., Romero, J. R., Ventura, S., & Data, Á. D. Á. (2014). *A Survey on pre-processing Educational Data Educational Data Mining*. (A. Peña-Ayala, Ed.) (Vol. 524). Cham: Springer International Publishing. http://doi.org/10.1007/978-3-319-02738-8

Romero, C., & Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, *33*(1), 135–146. http://doi.org/10.1016/j.eswa.2006.04.005

Romero, C., & Ventura, S. (2010). A Java desktop tool for mining Moodle data.

Romero, C., & Ventura, S. (2010). Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *40*(6), 601–618. http://doi.org/10.1109/TSMCC.2010.2053532

Romero, C., & Ventura, S. (2013). Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *3*(1), 12–27. http://doi.org/10.1002/widm.1075

Romero, C., Ventura, S., Espejo, P. G., & Hervás, C. (2008). Data Mining Algorithms to Classify Students. In *International Conference of Educational Data Mining*. Monteral, Canada.

Romero, C., Ventura, S., & García, E. (2008). Data mining in course management systems: Moodle case study and tutorial. *Computers & Education*, *51*(1), 368–384. http://doi.org/10.1016/j.compedu.2007.05.016

Rubinstein, R. Y., & Kroese, D. P. (2008). *Simulation and the Monte Carlo Method, 2nd Edition* (2nd ed.). Hoboken, New Jersey: John Wiley & Sons, Inc. http://doi.org/10.1111/j.1751-5823.2009.00074_8.x

Sachin, R. B., & Vijay, M. S. (2012). A survey and future vision of data mining in educational field. *Proceedings - 2012 2nd International Conference on Advanced Computing and Communication Technologies, ACCT 2012*. http://doi.org/10.1109/ACCT.2012.14

Sen, S. K., Dash, S., & Pattanayak, S. P. (2012). Agent Based Meta Learning in Distributed. *International Journal of Engineering Research*, *2*(3), 342–348.

Song, W., Wang, W., Xiong, M., & Jin, H. (2007). *Data Management Services in ChinaGrid for Data Mining Applications*. *Emerging Technologies in Knowledge Discovery and Data Mining* (Vol. 4819). http://doi.org/10.1007/978-3-540-77018-3

Srinivasan, L., & Treadwell, J. (2005). An Overview of Service-oriented Architecture, Web Services and Grid Computing Service-Oriented Architecture. *HP Software Global Business Unit Publication*.

Stankovski, V., Swain, M., Kravtsov, V., Niessen, T., Wegener, D., Kindermann, J., & Dubitzky, W. (2008). Grid-enabling data mining applications with DataMiningGrid: An architectural perspective. *Future Generation Computer Systems*, *24*(4), 259–279. http://doi.org/10.1016/j.future.2007.05.004

Stolfo, S. J., Prodromidis, A. L., Tselepis, S., Lee, W., Fan, D. W., & Chan, P. K. (1997). JAM: Java Agents for Meta-Learning over Distributed Databases. In *Knowledge Discovery and Data Mining* (pp. 74–81). Menlo Park, California: Pro- ceedings of the third international conference on knowledge discovery and data mining (KDD-97), AAAI Press.

Talia, D., & Trunfio, P. (2007). How distributed data mining tasks can thrive as services on garids. *Proceedings of National Science Foundation Symposium on Next Generation of Data Mining and Cyber-Enabled Discovery for Innovation.*, *53*(July), 132–137.

Talia, D., & Trunfio, P. (2012). *Service-Oriented Distributed Knowledge Discovery* (Vol. 20121229). 6000 Broken Sound Parkway NW, Suite 300: Taylor & Francis Group, LLC. http://doi.org/10.1201/b12990

Talia, D., Trunfio, P., & Verta, O. (2008). The Weka4WS framework for distributed data mining in service-oriented Grids. *Concurrency Computation Practice and Experience*, *20*(16), 1933–1951. http://doi.org/10.1002/cpe.1311

Tane, J., Schmitz, C., & Stumme, G. (2004). Semantic resource management for the web: an e-learning application. In *Proceedings of the WWW conference* (pp. 1–10). New York, USA. http://doi.org/10.1145/1013367.1013369

Thai-Nghe, N., Drumond, L., Krohn-Grimberghe, A., & Schmidt-Thieme, L. (2010). Recommender system for predicting student performance. *Procedia Computer Science*, *1*(2), 2811–2819. Retrieved from http://dx.doi.org/10.1016/j.procs.2010.08.006

Trčka, N., Pechenizkiy, M., & Aalst, W. V. D. (2011). Process Mining from Educational Data. In C. Romero, S. Ventura, M. Pechenizkiy, & R. S. J. D. Baker (Eds.), *Proceedings of Handbook of educational data mining,Chapman and Hall/CRC Data Mining and Knowledge Discovery Series* (pp. 123–142). Boca, Raton: Press, CRC.

Ueno, M. (2004). Data mining and text mining technologies for collaborative learning in an ILMS "Samurai." In *Proceedings - IEEE International Conference on Advanced Learning Technologies, ICALT 2004* (pp. 1052–1053). Joensuu, Finland. http://doi.org/10.1109/ICALT.2004.1357749

Umadevi, K., Maheswari, B. U., & Nithya, P. (2014). Design of E-Learning Application through Web, 5324–5329.

Vaghella, S. (2010). Weka Based Desktop Data Mining as Web Service, 692–710.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning and Techniques*. Elsevier.

Zafra, A., Romero, C., & Ventura, S. (2011). Multiple instance learning for classifying students in learning management systems. *Expert Systems With Applications*, *38*(12), 15020–15031. http://doi.org/10.1016/j.eswa.2011.05.044

Zafra, A., Romero, C., & Ventura, S. (2013). DRAL: A tool for discovering relevant e-activities for learners. *Knowledge and Information Systems*, *36*(1), 211–250. http://doi.org/10.1007/s10115-012-0531-8

Zaïane, O. R. (2001). Web Usage Mining for a better Web-Based Learning Environment. In *Proceedings of Conference on Advanced Technology for Education* (pp. 60–64). Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-72079-9_3

Zend Framework. (2015). Retrieved from http://framework.zend.com/

Zeng, L., Li, L., Duan, L., Lu, K., Shi, Z., Wang, M., … Luo, P. (2012). Distributed data mining: A survey. *Information Technology and Management*, *13*(4), 403–409. http://doi.org/10.1007/s10799-012-0124-y

Zorrilla, M., & García-Saiz, D. (2012). A service oriented architecture to provide data mining services for non-expert data miners. *Decision Support Systems*, *55*(1), 399–411. http://doi.org/10.1016/j.dss.2012.05.045

APPENDIX

Curriculum Vitae

Name Felermino Dário Mário António Ali

Date of Birth April, 3rd 1979

Education **Master of Science in Information Technology (by research)** - SEGi University, Malaysia. 2014-2016.

Bachelor's Degree in Computer Engineering - Universidade Lúrio - Lurio University, Mozambique. 2008-2012.

Publications "Moodle Data Retrieval for Educational Data Mining", International Journal of Scientific, Volume No.4 Issue No.10, pp: 523-525. (published in 1st November 2015)

"EDM Classifiers Performance and Comprehensibility Evaluation", proceedings of International Conference on Information Retrieval and Knowledge Management. (Accepted in 17 May 2016)