SMINER – a platform for data mining based on service-oriented architecture

Ahmed A.A. Esmin*, Denilson Alves Pereira, Marluc Rodrigues Pereira and Deivison Luiz Araújo

Department of Computer Science, Federal University of Lavras – UFLA, Campus Universitário – P.O. Box 3037, 37200-000, Lavras, MG, Brazil
Fax: +55-35-3829-1545
E-mail: ahmed@dcc.ufla.br
E-mail: denilsonpereira@dcc.ufla.br
E-mail: marluce@dcc.ufla.br
E-mail: deivison.araujo@gmail.com
*Corresponding author

Abstract: Data mining is a process to discover useful patterns in large volumes of data through the application of appropriate algorithms, tools, and techniques. However, building scalable, extensible, and easy-to-use data mining systems has proved to be a very difficult task. This paper presents the web platform called SMINER, which aims at interoperability and facility of integration in the development of data mining applications. The platform is based on the paradigm of service-oriented architecture (SOA) using the web services standard for extensibility and interoperability. This platform is composed of two main components: 1) web services, which implement data mining algorithms; 2) a user web interface, which can be used for modelling applications that use the services of the platform. Experimental results demonstrate that our SOA-based platform makes it easy to construct a flexible and scalable data mining system.

Keywords: service-oriented architecture; SOA; web services: data mining.

Reference to this paper should be made as follows: Esmin, A.A.A., Pereira, D.A., Pereira, M.R. and Araújo, D.L. (2013) 'SMINER – a platform for data mining based on service-oriented architecture', Int. J. Business Intelligence and Data Mining, Vol. 8, No. 1, pp.1–18.

Biographical notes: Ahmed A.A. Esmin is a Full Professor at the Department of Computer Science, Federal University of Lavras (DCC-UFLA), Minas Gerais, Brazil, where he directs the Computational Intelligence and Advanced Systems (LICESA) lab. His research is in computational intelligence, swarm intelligence, machine learning, data mining, text mining and their applications. He is author and co-author of more than 65 research papers. He is currently working as Visiting Researcher at the Text Analysis and Machine Learning (TAMALE) lab, School of Electrical Engineering and Computer Science, University of Ottawa (uOttawa), Canada. He is also a programme committee member for a number of international conferences in AI and machine learning. He served as a reviewer for the leading journals in his field.
Denilson Alves Pereira is a Professor of Computer Science at the Federal University of Lavras, Brazil, acting in undergraduate and graduate programmes. His research interests include information retrieval, data mining, and databases. He received his doctor degree in Computer Science from the Federal University of Minas Gerais, Brazil, in 2009.

Marluce Rodrigues Pereira is a doctor in Computer and Systems Engineering by Federal University of Rio de Janeiro (UFRJ), Brazil. She is a Full Professor at the Department of Computer Science, Federal University of Lavras (DCC-UFLA), Minas Gerais, Brazil, where she is a contributor of Computational Intelligence and Advanced Systems (LICESA) laboratory. Her research is focused in parallel and distributed processing and intelligent systems.

Deivison Luiz Araújo is a graduate student at the Department of Computer Science, Federal University of Lavras (DCC-UFLA), Minas Gerais, Brazil and a member of the Computational Intelligence and Advanced Systems (LICESA) lab. His research is in computational intelligence and data mining.

1 Introduction

A major challenge of computer science is to develop solutions for the treatment, retrieval, and dissemination of relevant information from exponentially growing volumes of data. To interpret and examine such data in an automatic and intelligent way, new tools and techniques are required.

From this necessity arises data mining, a nucleus of the complex process of knowledge discovery in databases (KDD) (Han et al., 2011), defined as the application of algorithms for knowledge extraction in databases. The application of these algorithms becomes a computationally expensive task when it comes to large databases, which requires the need for tools that make such executions faster.

In addition, if for each new application, it was necessary to implement the algorithms for data mining again, very repetitive work would be done. Even with software reuse techniques, various adjustments related to languages and platforms would be needed.

To resolve this problem of rework, techniques are being developed to make possible to build systems that have components distributed over the web, and are independent of programming languages. A new architecture, called service-oriented architecture (SOA) (Perrey and Lycett, 2003), for software development has been proposed. SOA has as main objective the intense reuse of components (services). Thus, the task of developing an application is primarily to develop the composition and coordination of services already implemented, increasing the reuse and reducing the expenditure of resources.

One of the first proposals for work on providing data mining services on the Internet was made by Sarawagi and Nagaralu (2000). Since then, much research around the topic has been made. Guedes et al. (2006) propose a platform that aim at scalability, extensibility, simple abstraction for users, and high performance. Wu et al. (2007) describe a SOA for business intelligence (BI) that integrates BI technologies in a single environment, simplifying data traffic and allowing low latency analysis. Li and Song (2010) address the integration of data mining web services (WS) using SOA. This integration uses the Java Data Mining (JDM), which is the first attempt to create a
standard Java application programming interface (API) to access mining data tools from Java applications. Talia and Trunfio (2010) discuss how technology for data distribution can give support for data mining applications and services and for knowledge discovery. They highlight their study involving data distribution using the grid computing paradigm. Chan et al. (2011) developed a conceptual service-oriented BI architecture model based on their studies of existing ones. Wang et al. (2011) developed a financial data mining system based on SOA, which provides data analysis based on data warehouse. Zorrilla and García-Saiz (2012) proposed a SOA implemented by means of WS with the aim of helping non-expert data miners to extract useful and novel knowledge using data mining techniques.

This paper presents SMINER, a platform for developing of data mining applications, which aimed at interoperability and ease of integration with other applications. The platform was developed based on the SOA paradigm, using the standard WS for extensibility and interoperability, and offers simple abstractions for users. It makes available an implementation of the particle swarm optimisation for cluster data (CPSO) algorithm (Esmin et al., 2008; Cura, 2012), the scalable distributed parallel apriori (DPA) algorithm (Kun-Ming et al., 2010), and the main classic data mining algorithms from the Waikato Environment for Knowledge Analysis (WEKA) library (Witten et al., 2011), as a basis for implementation of data mining algorithms. It also uses the AXIS2 framework as a Simple Object Access Protocol (SOAP) implementation for WS and the Google Web Toolkit (GWT) for developing the user interface. The platform architecture as well as the steps needed to create an example application is presented, demonstrating how to build a data mining application using SOA with WS. In addition, an experimental result of using DPA and an example application using the platform is presented.

The paper is organised as follows. A brief description of the data mining process is reviewed in Section 2. The concepts of the service oriented architecture are described in Section 3. The SMINER platform is presented in Section 4. An example to illustrate the use of the proposed platform is given in Section 5. The experimental results are shown in Section 6. Finally, the conclusion is stated in Section 7.

2 An overview on data mining

According to Han et al. (2011), we are drowning in data, but dying of thirst for knowledge. That is, the need for turning such data into information and knowledge useful for decision support is imminent, which has demanded considerable investments from the scientific community and the software industry.

Therefore, a process called KDD was developed, which according to Fayyad et al. (1996) is the non-trivial process of identifying in data, patterns that are valid, novel, potentially useful, and understandable, in order to improve the understanding of a problem or a decision-making procedure.

Data mining, the focus of this study and part of the overall process of KDD, consists of a set of techniques gathered from statistics, database, and artificial intelligence with the aim of discovering new, useful, relevant, and non-trivial knowledge that can be hidden in a large body of data (Han et al., 2011).

The data mining techniques consist of specification of methods to find patterns of interest of organisations. For this, there are several types of data mining algorithms such
as association rules, clustering, and classification, among others. Several tools implement these mining algorithms; an example is the platform WEKA (Witten et al., 2011; University of Waikato, 2011). The WEKA tool is a package developed at the University of Waikato in New Zealand, to add algorithms for data mining in the area of Artificial Intelligence.

WEKA supports many data mining tasks, more specifically, preprocessing, clustering, classification, regression, visualisation, and feature selection (Hall et al., 2009). WEKA also provides access to Structured Query Language (SQL) databases using Java Database Connectivity (JDBC) and can process the result returned by a query to the database. In addition, the software is licensed under the GNU General Public License, therefore it is possible to change the source code developed in Java language.

3 Service oriented architecture

3.1 Concept

According to Mackenzie et al. (2006), SOA is a paradigm for organising and use of capabilities that are under the control of different ownership domains.

To understand SOA, it is important to understand the concept of service. According to Mitra (2003), service is an abstract resource that represents the ability to perform tasks that have a consistent feature in the view of the provider and consumer. In SOA, the services are software components that publish their interfaces of use, and which are independent of an operating system, language, or platform. That is, they are interoperable, the most important principle in SOA, according to Guedes et al. (2006) and Jyotishman et al. (2008).

Among the advantages of using SOA, the most important ones that led to its use in this study are: easy adaptation of applications to emerging technologies, high encapsulation of functionality, ease of creating new services by composing from existing services, easy integration with other applications systems, reuse and reduction of investment costs in organisations for cooperation among them.

The SOA architecture consists of some main elements: the actual services is composed of the contract of accession, interface, and implementation that involves business logic and access to the database, the repository where services are available, the protocol for exchanging messages and client applications that make use of available resources for the services architecture.

The key point to develop a system in SOA is to define the standard of communication that the services will be created with. There are some ways to implement SOA, but the best and most commonly used are those with WS, according to Li and Song (2010). Nevertheless, SOA is more than just a set of technologies and is not directly related to any technology.

Another important factor to consider is the granularity of the service. Coarse-grained services include several features that offer clients a greater abstraction of the problem. Fine-grained services are more specific and practically represent methods of access to resources, leaving to the consumer the implementation of the logic and the control of the interaction between them.
3.2 Web services

WS are a standard way of communication between different software applications running on a variety of platforms and/or frameworks. WS are characterised by their great interoperability and extensibility (Chinnici et al., 2007). The goal is to develop software or components capable of interacting with other software, making them open and independent of any particular language or platform. Therefore, it is necessary to define a set of structured data formats that are readable by humans.

WS have emerged as an innovative proposal, as an evolution of some models of distributed computing [remote method invocation (RMI), distributed component object model (DCOM), common object request broker architecture (CORBA)]. In addition to defining interfaces for software components and methods to access them through standard and interoperable protocols, it provides mechanisms for the discovery of new services. One of the major gains from this is the possibility of creating new applications and services by composing and combining existing services.

To implement services using WS based on SOA principles – which are independent of any architecture, easy processing, easy access, and has wide acceptance – some elements should be considered: a data transmission protocol widely adopted, a data representation format independent of architecture, a way to clearly express the location, functionality and access interface of each service, and a protocol to express the semantics of the process of requesting services and obtain answers. The main elements to be considered in a web service and the technologies used to implement them are described in the following paragraphs and can be seen in Figure 1.

**Figure 1** The main elements of a web service (first column) and the technologies used to implement them (second column) (see online version for colours)

For the transport element, according to Mulligan and Gracanin (2009), the Hyper Text Transfer Protocol (HTTP) is the most suitable protocol to be used as a channel for transferring data in a distributed environment with WS.

For the message element, the format of data representation is an important problem to be solved in SOA. It is necessary to have a well-defined format so that there is universal understanding of the messages exchanged, regardless of platform or language. The commonly used format is Extensible Markup Language (XML) (Ya-Qin and Yong-Wen, 2010). The main reason for its use in WS is due to its high portability. Although HTTP is responsible for the transfer of resources on the web, you need a protocol that can send and receive messages containing XML documents; this can be done by SOAP...
SOAP is able to completely describe a remote procedure call with its parameters, even the most complex data structures.

For the next element, the service description, publication, and location are important factors to be considered when creating a WS. For this, the Web Service Description Language (WSDL) (W3C WSDL, 2007) was created. The WSDL has a text format based on XML, formed by two different parts: one abstract and one concrete. The abstract part describes the interface of services – defines operations, parameters and return – that is, describes how the service can be invoked by clients. The concrete part defines the communication protocol and the address where the service will be available.

For the last element, to find service descriptions, it is necessary to publish them in some repository of WS. Normally, such descriptions are implemented using the standard Universal Description, Discovery and Integration (UDDI) (Papazoglou and Georgakopoulos, 2003). The UDDI standard defines a protocol for implementation of directories for WS registries, where are also available the information about the companies that provide services. Its implementation consists of several WS, which provide an interface to allow clients to interact with information stored there.

### 3.3 SOAP and the AXIS2 framework

SOAP is the standard protocol for transmitting data within the WS architecture proposed by W3C. It is a protocol based on XML and follows the ‘request-response’ model of HTTP (Chappell et al., 2002; Zhuge and Liu, 2004). The process is described by Meng et al. (2009) as follows: first, the publication of the WSDL in a UDDI repository is made; subsequently, the client makes a search in the service repository, downloads the WSDL descriptor, and finally performs the XML invocation of the service via SOAP protocol and receives an XML response of the required processing (Di Penta et al., 2008).

Apache AXIS2 (Apache, 2011) is a SOAP implementation used for construction of WS. AXIS2 is an open source framework based on Java and XML. Is has a small and simple application server. It is also possible through the AXIS2 to automatically generate the WSDL file from a Java interface, and vice versa. In addition to the Java version, there is a version based on C++.

Despite using the Java version for AXIS2, SOAP is not related to any language. It can be implemented in any language according to the standards of the WS. The choice of Java as the development language was driven by ease of using AXIS2, along with the use of the WEKA library, as both are already written in Java.

### 4 The SMINER platform

#### 4.1 Overview

The SMINER platform aims to provide WS for scalable knowledge mining in databases, specifically in the process of applying data mining algorithms. It was implemented as WS the main data mining algorithms from the WEKA library, the cluster particle optimisation (CPSO) based on Esmin et al. (2008), Cura (2012) and Belal et al. (2004) and a scalable implementation of the DPA algorithm based on Kun-Ming (2010). The KDD tasks such as collecting, cleaning, and preprocessing of data were not included.
The platform is composed of two main components: the WS, which implement data mining algorithms and a web interface with the user, which can be used for modelling applications that use the services of the platform. Figure 2 illustrates the architecture of the SMINER Platform.

**Figure 2** Architecture of the SMINER platform (see online version for colours)

The component responsible for the services is available on the web and can be found through the UDDI protocol, where the WSDL contracts for its use are registered. For the exchange of information between application and WS, the SOAP protocol is used. With this well-defined standardisation, any developer can easily use the services.

The web user interface is an option with which users can experiment with algorithms and create models without the need to develop an application. This interface uses extensively all services of the platform, in addition to their composition, using them as a single service.

In Talia et al. (2008), the authors present the Weka4WS, a framework that extends the WEKA toolkit to support distributed data mining on Web Services Resource Framework (WSRF)-enabled grids. According to the authors, the Weka4WS adopts the WSRF technology for running remote data mining algorithms and managing distributed computations. The goal of Weka4WS was to extend WEKA to support remote execution of the data mining algorithms. This makes distributed data mining tasks to be concurrently executed on decentralised grid nodes by exploiting data distribution and improving application performance.
However, the main difference between the SMINER and the Weka4WS is related to the approach; whereas the Weka4WS aims to improve the performance and execution in grids, the SMINER aims to make the data mining algorithms as a service that run on scalable environment and the possibility of the integration of other data mining algorithms that are not related to the WEKA. Both approaches are possible and do not conflict and they can be integrated to complement each other.

Besides, SMINER being constructed as WS, it is transparent for the user what is behind the scene, in the server-side. The efficiency of the algorithms can be improved by adopting new technologies such as Hadoop/MapReduce (Dean and Ghemawt, 2008; Lin and Kolcz, 2012). SMINER can also contribute to the development of intelligent data management applications such as in Sakurai and Tsutsui (2011).

4.2 Features

The features offered by the SMINER platform take into account the whole process of applying data mining algorithms, from the transfer of data, choice of algorithm, and its configuration options, up to the execution and achievement of results.

The data to be used by the algorithms are uploaded and stored in a database on the web server of the platform, and can be reused later, until the user/developer owner chooses to discard them.

Likewise, the user/developer has the option to create models and save them for later use.

The platform offers algorithms related to techniques of association rules, classification, and clustering. The user/developer chooses the algorithm to be executed, along with its configuration options.

To meet the features that depend on save and reuse models, and data and results from a particular user/developer, the use of the service requires identification. For this, a user identification procedure was built into the system. The user/developer must make a prior registration and login/logoff when using the system.

4.3 Services

The platform features are available through WS. Each service includes a feature and the set or composition of such services forms the platform for data mining. Services were set so that each one treats of an independent feature and uses a minimum of data, thus improving response time to a set of services.

The services are provided using a fine-grained model, allowing the user/developer greater freedom in managing the interaction between each service. Table 1 presents the available services and a brief description of them.
Table 1  Offered services

<table>
<thead>
<tr>
<th>Offered services</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>String register (login, password, name, surname, e-mail)</td>
<td>Registers a user on the system returning a message or an error code.</td>
</tr>
<tr>
<td>User login (login, password)</td>
<td>Logs in and returns the user.</td>
</tr>
<tr>
<td>String[] showSavedFiles (user)</td>
<td>Returns the names of the user data files.</td>
</tr>
<tr>
<td>String uploadFile (fileName, base64file, user)</td>
<td>Sends a data file in base64 format, and returns a message of confirmation or an error code.</td>
</tr>
<tr>
<td>String discardFile (fileName, user)</td>
<td>Discards the data file selected by the user. Returns a message of confirmation or an error code.</td>
</tr>
<tr>
<td>String[] getAlgorithms (user)</td>
<td>Returns the names of supported algorithms.</td>
</tr>
<tr>
<td>Algorithm getAlgorithm(algorithmName, user)</td>
<td>Returns the selected algorithm.</td>
</tr>
<tr>
<td>String[] getOptionsDesc (algorithmName, user)</td>
<td>Returns the description of each option of the selected algorithm.</td>
</tr>
<tr>
<td>String jointOptions (options, user)</td>
<td>Joins the options of the algorithm into a String.</td>
</tr>
<tr>
<td>String associator (algorithm, fileName, user)</td>
<td>Performs an association algorithm. Returns a string containing the result of mining.</td>
</tr>
<tr>
<td>String classifier (algorithm, fileName, option, fileName2, percent, user)</td>
<td>Performs a classification algorithm. Returns a string containing the result of mining.</td>
</tr>
<tr>
<td>String clusterer (algorithm, fileName, option, fileName2, percent, user)</td>
<td>Performs a clustering algorithm. Returns a string containing the result of mining.</td>
</tr>
<tr>
<td>String saveModel (user)</td>
<td>Saves the last model created by the user. Returns a message of confirmation or an error code.</td>
</tr>
<tr>
<td>String discardModel (model, user)</td>
<td>Discards the model selected by the user. Returns a message of confirmation or an error code.</td>
</tr>
<tr>
<td>String useModel (fileName, model, user)</td>
<td>Uses a saved model on a data file. Returns the result of mining.</td>
</tr>
<tr>
<td>Model[] listModels(user)</td>
<td>Returns a list of saved models of a given user.</td>
</tr>
</tbody>
</table>

4.4  Web user interface

The web interface is an alternative for users to use the services of data mining platform SwamMineWeb without the need to write a code in a programming language. It provides a friendly interface with simple abstractions for algorithms implemented on the platform.

Figure 3 presents a snapshot of the interface to run the clustering algorithm called Simple K-means. The user can send a data file, set the desired options of the algorithm, process it, and obtain the results. To take advantage of the application, the user only needs to register. The left side of the screen presents the algorithms available to the user.
4.5 Implementation

For implementation of WS, Java language was used. It was chosen because of its maturity in building WS and for having tools that facilitate the development process. However, the use of Java does not lose the characteristics of interoperability, that is, the possibility of using the services in other languages.

WS were created using the AXIS2 framework. This framework incorporates a SOAP server that operates with the Tomcat Web Server. It also has features to make a Java class in a WSDL descriptor, and turn objects into XML files automatically, and support C, C++ and Java.

The algorithms: CPSO and the DPA were implemented in C++ using the message passing interface (MPI) libraries (http://www.mcs.anl.gov/MPI and OpenMP, http://www.openmp.org).

The interface was made using the framework GWT. Its code, written in Java and compiled to optimised JavaScript, adds Web 2.0 features, with dynamic content, and is compatible with most browsers currently available.

The SMINER was executed in a PC cluster with 8 computing nodes. Table 2 gives the hardware and software specifications.
Table 2  Hardware and software specifications

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Intel Core 2 Duo 2.66 GHz frequency and 2 GB of RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network</td>
<td>Intel Corporation 82566 DM-2 Gigabit Ethernet</td>
</tr>
<tr>
<td>Disk</td>
<td>160 GB of hard disk</td>
</tr>
<tr>
<td>The operating system</td>
<td>Gentoo Linux, kernel 2.6.38.6, 32 bits</td>
</tr>
<tr>
<td></td>
<td>OpenMPI version 1.4.2</td>
</tr>
<tr>
<td></td>
<td>gcc version 4.4.5</td>
</tr>
</tbody>
</table>

5 Creating client applications

This section describes step-by-step on how to develop client applications that use the WS of the platform SMINER using Java language.

When using NetBeans 6.8 as a development environment, the necessary plugins must be installed first. The three plugins used were ‘AXIS2 Support’, ‘GWT4NB’, and ‘WADL Designer’.

After installing the plugins, the desired project must be created, either web-based or desktop. In the following example, a Java desktop project was created. That done, click on the project with the right mouse button and select ‘New/Other’. In the ‘Categories’ window, select ‘Web Service’, and in ‘File Types’ select ‘Web Service Client’ as shown in Figure 4.

Figure 4  Creating a client to web service (see online version for colours)
In the next window, provide the address of the WSDL file, select the client's style as 'JAX-WS Style', and confirm the settings of the service, as shown in Figure 5.

**Figure 5** Window for setting services (see online version for colours)

With the client access to the web established, the service will automatically generate the classes required for using the platform. Finally, classes should be created to utilise the services, as the example shown in Figure 6.

**Figure 6** Sample code (see online version for colours)

```java
  1  import org.smws.Services;
  2  import org.smws.ServicesPortType;
  3  import org.smws.model.xsd.Algorithm;
  4  import org.smws.model.xsd.User;

  5  public class ExampleAssociate {
  6      public static void main(String[] args) {
  7          Services serv = new Services();
  8          ServicesPortType ws = serv.getServicesHttpSampleEndpoint();
  9          User u = ws.login("user", "password");
 10          Algorithm alg = ws.getAlgorithm("weka.associations.Apriori", u);
 11          String print = ws.associate(alg, "weather.nominal.arff", u);
 12          while (!print.contains("<BR>")) {
 13              print = print.replaceAll("<BR>", "\n");
 14          }
 15          System.out.println("Results:
" + print);
 16      }
 17  }
```
In this example, in lines 1–4, imports of classes that will be used are made. They are part of the classes automatically generated when creating a client for a web service, as described earlier. In lines 8 and 9, an instance of the class ‘ServicosPortType’ is created, which provides access to the services of the platform. In line 11, login in the system is established, passing as parameters the user login and password, and returning the User object, which will be used for other services. After logging, some algorithms must be chosen from those available. In this example, the apriori algorithm was chosen, with its default parameters, as shown in line 12. To perform the data mining itself, any of the
three methods available can be used: associator, classifier, or clusterer. To use the apriori algorithm, the method to be accessed is the associator (line 13), passing as parameter the algorithm returned earlier and the data file name. In this case, it is considered that the data file has been sent to the system. The result of running the algorithm is returned in the form of a string with HTML tags. In the example, the result is printed to standard output (lines 14–17), as shown in Figure 7.

6 Experimental results and application

6.1 Experimental results

Experiments were made to evaluate the performance of SMINER by using as WS the DPA algorithm (SMINER DPA) and compared with a DPA stand-alone implementation (Std DPA). The synthesised datasets generated by IBM’s Quest Synthetic Data Generator (IBM Almaden, 1994) were used to compare these two different configurations. This dataset (T10I4D6KN100K) consist of 800,000 transactions, 100,000 items available and with an average of 10 items per transaction. The minimum support (Sup) value used was 0.0015. The SMINER DPA was executed for 10 times in a PC cluster with 8 computing nodes (as specified in Table 2).

Figures 8 and 9 illustrate the average of the execution time and the speed-up by various numbers of processors, respectively. With more processors, SMINER DPA needed less execution time and shows better performance than the Std DPA.

Figure 8 Execution time of Std DPA and SMINER DPA for a large dataset (see online version for colours)
Figure 9  Speed-up ratios of SMINER DPA for a large dataset (see online version for colours)

Figure 10  An application that uses SMINER (see online version for colours)
6.2 An experimental application

To experiment the use of the platform, we start the development of an application for opinion mining and sentiment analysis to analyse the reaction of fans during a soccer match in Brazilian championship using the Twitter as source (i.e. match between fans on twitter before, during and after the games!). All the classification process is done as service (SOA) provided by the SMINER platform. Figure 10 shows the application under development.

7 Conclusions and future work

This study aimed to present the development of a data mining platform for easy integration with other applications. The main motivation was the growing interest in many different areas of knowledge to find interesting patterns in their databases. To realise such interesting patterns, the task of mining should be as efficient as possible in terms of computational time.

It can be concluded that the objectives were met satisfactorily. The platform was developed as a web application in the client/server model, so that the user can take advantage of an online platform. Thus, the application of complicated data mining techniques can focus on servers with high processing power, with the users' machines doing the job of sending data, the presentation of results, analysis, among others.

To provide the data mining tasks as WS, we studied the feasibility of the paradigm of SOA. Hence, a conclusion was reached that SOA would add features such as ease of service composition, reusability, and especially interoperability. WS demonstrated to be proficient to implement SOA, as they provided all requirements.

The platform created in this study, the SMINER, was developed using some highly diffused tools. For data mining, we used the WEKA platform, which is a free tool, open source, and with implementations for several algorithms and other scalable algorithms (DPA) and Cluster PSO. For the creation of WS, AXIS2 library was used, which proved to be powerful and easy to use, besides being free, and is maintained by the Apache Software Foundation.

The modelling of SMINER along with the experience of using the tools discussed, demonstrated the great potential and ease of use of WS as the embodiment of a SOA, concluding that SOA is a good alternative for building web systems aimed primarily at high performance and reuse of software regardless of language and architecture.

As future study, we intend to develop a scalable version of WEKA library and integrate it with other data mining algorithms, as well as, analyse the performance of the platform using large applications. We are working on improving the efficiency of data mining algorithms by implementing them in the framework Hadoop/MapReduce.

Acknowledgements

The authors would like to thank CNPq and FAPEMIG (Brazilian agencies) for partial financial support.
References


