KEEL: UNA HERRAMIENTA DOCENTE PARA SISTEMAS DIFUSOS

Joaquín Derrac[1], **Julián Luengo**[2], **Alberto Fernández**[3], **Salvador García**[3], **Jesús Alcalá-Fdez**[1]

[1]*Departamento de Ciencias de la Computación e Inteligencia Artificial,CITIC-UGR, Universidad de Granada, 18071 Granada, Spain*, {jderrac,jalcala}@decsai.ugr.es

[2]*Departamento de Ingeniería Civil, EPS, Universidad de Burgos, 09006, Burgos, España*, jluengo@ubu.es

[3]*Departamento de Ciencias de la Computación, Universidad de Jaén, 23071 Jaén, España*, {alberto.fernandez,sglopez}@ujaen.es

Resumen

En la actualidad, los sistemas difusos están cobrando cada vez más relevancia en los planes de estudio de asignaturas de minería de datos y aprendizaje automático. En este contexto, las herramientas software de código abierto ofrecen a los estudiantes la posibilidad de experimentar con las técnicas abordadas, reforzando así su aprendizaje. Entre ellas, destacamos KEEL como una herramienta versátil para evaluar y analizar el funcionamiento de los algoritmos estudiados.

El objetivo de este trabajo es presentar la faceta educativa de la herramienta software KEEL: Un módulo educativo de utilidad para reforzar la docencia en asignaturas de introducción al aprendizaje automático y a los sistemas difusos. Este módulo proporciona al usuario la capacidad de visualizar el progreso de los algoritmos, ayudándole a evaluar, ajustar y comprender el funcionamiento de las técnicas clásicas del área.

Palabras Clave: Minería de Datos, Aprendizaje Automático, Sistemas Difusos, Java, Herramienta Software, Software Educativo.

1 Introducción

En los últimos años es habitual la inclusión de asignaturas de Minería de Datos y Aprendizaje Automático como parte de la formación de los estudiantes de Ciencias de la Computación. Este tipo de materias, cruciales para su formación científica [11], les ayudan a obtener una serie de conceptos claves en su formación como profesionales, aportándoles importantes herramientas para el futuro.

Pese a ello, el aprendizaje de las técnicas derivadas de estos campos no siempre es sencillo. Este aprendizaje suele requerir habilidades de análisis que permitan al estudiante entender las características más importantes de cada técnica. En el Aprendizaje Automático, el problema subyacente consiste en que la implementación de la mayoría de los métodos relacionados requiere de ciertos conocimientos de programación, así como de una cantidad considerable de tiempo y esfuerzo. Así, se corre el riesgo de acabar empleando las horas dedicadas a la enseñanza de esta materia como meras *clases de programación*, en lugar de centrarse en el análisis de las características de los algoritmos, desde un punto de vista teórico y práctico.

Una manera de aliviar este problema consiste en emplear herramientas software que incluyan dichas implementaciones. Actualmente es posible encontrar en la Web un número elevado de ellas. Sin embargo, la mayoría están destinadas a abordar una determinada aplicación - siendo difíciles de aprovechar desde un punto de vista docente.

Sin embargo, en los últimos años han aparecido cierto número de herramientas de propósito general. Aunque muchas son comerciales, unas pocas (como Weka [5] o JavaML [1]) son distribuidas de forma libre, alcanzando gran popularidad dentro de la comunidad investigadora gracias a su condición de *software libre* [12] [1]).

KEEL [2, 3] es una herramienta software libre desarrollada completamente en Java. KEEL permite al usuario emplear una gran cantidad de técnicas de Aprendizaje Automático en diferentes tipos de problemas: Regresión, clasificación, agrupamiento, asociación, etc., incluyendo una gran recopilación de los Sistemas Difusos existentes. Además de como herramienta para investigación, KEEL ha sido diseñado también con características educativas.

En este trabajo se describen los principales aspectos de KEEL como herramienta docente. Su módulo educativo es presentado como una herramienta capaz de ofrecer una visión del funcionamiento de los algoritmos, permitiendo al estudiante comprender y adaptar su funcionamiento. Así,

[1](Más información en los siguientes enlaces: KDnuggets software http://www.kdnuggets.com/software The-Data-Mine site http://the-data-mine.com/bin/view/Software

esta herramienta puede usarse como apoyo práctico en todo tipo de asignaturas de Minería de Datos y Aprendizaje Automático, incluyendo aquellas con una fuerte componente de técnicas basadas en lógica difusa.

Es importante destacar que esta herramienta ya ha sido utilizada con éxito en varios programas de formación, tanto nacionales como internacionales. Por ejemplo, recientemente se empleó como apoyo al proyecto MIBISOC (Medical Imaging using Bio-Inspired and Soft Computing) [2].

El resto del trabajo se organiza como sigue: la Sección 2 describe brevemente las características generales de KEEL. La Sección 3 da una visión del módulo educativo, destacando sus capacidades más sobresalientes. La Sección 4 presenta un caso de uso considerando el análisis de varios Sistemas Difusos clásicos. Finalmente, en la Sección 5 se presentan las conclusiones del trabajo.

2 Descripción de KEEL

KEEL es una herramienta software para la preparación de algoritmos de minería de datos. La versión actual de KEEL está compuesta por los siguientes módulos (ver Figura 1):

Figura 1: Pantalla principal de KEEL

- *Tratamiento de datos* (Data Management): Este módulo contiene una serie de herramientas de tratamiento de datos: Importación, exportación, edición y visualización de datos, aplicación de transformaciones, etc.
- *Experimentos* (Experiments): Este módulo está dedicado al diseño de experimentos, proporcionando numerosas opciones: Tipo de validación, tipo de aprendizaje (clasificación, regresión, aprendizaje no-supervisado), etc.
- *Educativo* (Educational): Este módulo permite realizar experimentos interactivos. Con una estructura similar al módulo anterior, permite diseñar experimentos con propósitos educativos.
- *Módulos* (Modules): Adicionalmente, KEEL incluye un módulo para datos no-balanceados [4], un módulo de análisis estadístico no paramétrico [7, 9], y un módulo de aprendizaje multi-instancia [8].

Esta estructura hace que KEEL sea una herramienta interesante para distintos tipos de usuarios. Las principales características de KEEL son las siguientes:

- Posee una librería estadística para análisis de algoritmos. Los tests de esta librería permiten analizar la bondad de los resultados obtenidos, realizando comparaciones paramétricas y no paramétricas.
- Incluye algoritmos de aprendizaje de modelos predictivos, de preprocesamiento (discretización, selección de instancias, selección de características, etc.) y post-procesamiento. También incluye muchas propuestas del estado del arte de diferentes áreas de la Minería de datos, como por ejemplo árboles de decisión, sistemas difusos basados en reglas, etc.
- Ofrece al usuario una interfaz amigable, orientada al análisis de algoritmos.
- Permite crear experimentaciones conteniendo múltiples conjuntos de datos y algoritmos conectados entre sí. Los experimentos son generados mediante scripts independientes de la interfaz de usuario, para permitir una ejecución separada en la misma u otras máquinas.

Para más información sobre las características de la herramienta KEEL, pueden consultarse los trabajos [2, 3] o el sitio web del proyecto KEEL (`http://www.keel.es`).

3 Módulo Educativo

En el ámbito docente, las necesidades de un estudiante son muy diferentes de las de un investigador. Generalmente, sus necesidades no consisten en realizar experimentaciones con un gran número de problemas y métodos del estado del arte. Generalmente, los experimentos a realizar durante el aprendizaje serán más sencillos, orientados a comprobar la evolución y los resultados de los algoritmos más comunes de forma clara.

Siguiendo esta idea, el módulo educativo de KEEL ha sido diseñado para dar cabida a las propuestas más representativas de cada área. Así, ofrece las técnicas de preprocesamiento y aprendizaje más populares, junto con una amplia colección de problemas representativos en los ámbitos de clasificación y regresión.

[2] MIBISOC es una Red de Entrenamiento Internacional del programa Marie Curie International Training Network soportada por la Comisión Europea dentro del Séptimo Programa Marco (FP7 PEOPLE-ITN- 2008). http://www.softcomputing.es/mibisoc/

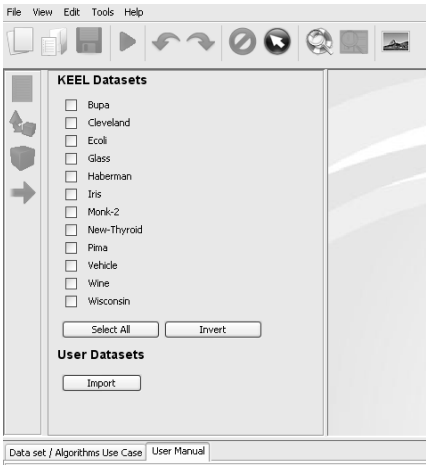

Figura 2: Selección de conjuntos de datos. Es posible añadir nuevos conjuntos, mediante el botón de importar.

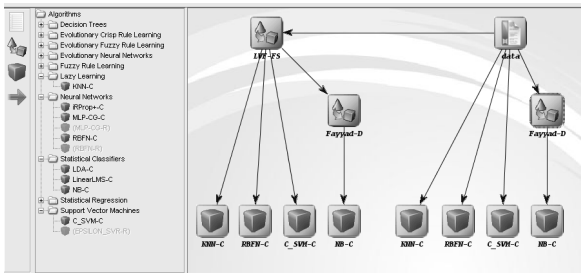

Figura 3: Un experimento para analizar el rendimiento del Filtro Las Vegas de selección de características sobre 4 clasificadores: K-NN, Naive Bayes, una red neuronal de base radial y una máquina de vectores soporte. Además se ha empleado el discretizador de Fayyad para adecuar los datos en el caso de Naive Bayes.

A la hora de crear un nuevo experimento con el módulo educativo, el primer paso consiste en seleccionar el tipo de problema (clasificación o regresión) y el esquema de validación a emplear (validación cruzada de k- particiones, validación cruzada 5x2 o sin validación). Una vez determinados ambos puntos, deben indicarse qué conjuntos de datos se desean emplear. La Figura 2 muestra un ejemplo en el contexto de clasificación, donde es importante destacar que el usuario puede incluir nuevos conjuntos mediante el botón de importar (*Import*).

La interfaz principal permite el diseño de experimentos de forma gráfica. La Figura 3 muestra un experimento diseñado para evaluar el comportamiento de varios clasificadores bajo conjuntos de datos preprocesados con un método de selección de características. Arrastrando y colocando los iconos que representan a cada técnica, los estudiantes pueden diseñar fácilmente este tipo de experimentos sin necesidad de emplear complicados procedimientos para establecer los parámetros, algoritmos y conjuntos de datos a utilizar típicos de un diseño experimental mayor.

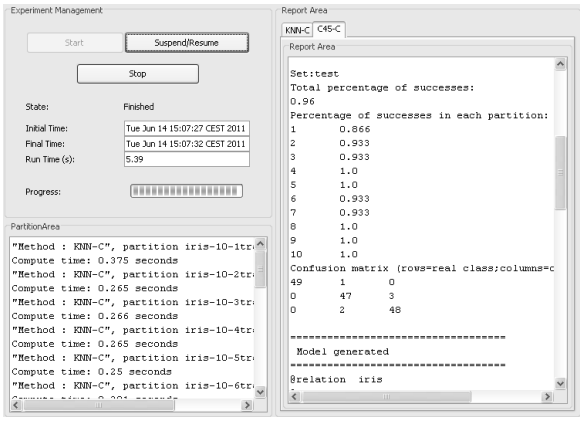

Figura 4: Ventana de gestión de experimentos. En ella se muestran los resultados obtenidos, los tiempos de ejecución y los modelos generados.

Esta facilidad de uso es especialmente importante si consideramos que el módulo educativo contiene una amplia colección de métodos pertenecientes a áreas muy diferentes entre sí. Por ello, disponer de un modelo sencillo para crear y configurar los experimentos que los incluyan es crucial, puesto que no se puede exigir a los estudiantes en estas etapas de su formación que adquieran un conocimiento profundo de cada una de las técnicas para poder emplearlas.

En el módulo educativo, el experimento se desarrolla en tiempo real. Una vez creado, el usuario puede escoger entre ejecutarlo o guardarlo en formato XML. Si se ejecuta, el sistema mostrará una ventana auxiliar para gestionar y visualizar el desarrollo del experimento (Figura 4). Una vez finalizado, esta ventana mostrará los resultados obtenidos por cada algoritmo, incluyendo información adicional como, por ejemplo, las matrices de confusión en clasificación o los errores medios obtenidos en regresión.

De esta manera, el usuario dispone de una visión del progreso de los algoritmos, pudiendo acceder a ellos directamente desde la propia interfaz. En el área de resultados se muestran los modelos generados, de gran utilidad como panorámica del conocimiento extraído a partir de los datos y su estructura intrínseca (por ejemplo, los algoritmos de árboles de decisión mostrarán una representación de los árboles construidos, mientras que para los sistemas basados en reglas se obtendrá la base de reglas generada).

Gracias a estas características, el módulo educativo es ideal para estudiantes que deseen analizar y mejorar los resultados obtenidos por un algoritmo dado. Los estudiantes pueden usarlo para comparar varios métodos relacionados sobre un conjunto fijo de problemas, e incluso para tratar de descubrir qué algoritmos o metodologías son más eficaces de cara a un problema concreto. Mediante la información obtenida en los informes de resultados, pueden tratar de determinar las causas del buen o mal comportamiento de un método para, más adelante, tratar de mejorarlo (adaptando, por ejemplo, la configuración de parámetros empleada).

Tabla 1: Conjuntos empleados en el estudio

Conjunto	# Ej.	# At.	# Cl.	Conjunto	# Ej.	# At.	# Cl.
Bupa	345	6	2	Monk-2	432	6	2
Ecoli	336	7	8	New-Thyroid	215	5	3
Glass	214	9	7	Pima	768	8	2
Haberman	306	3	2	Vehicle	846	18	4
Iris	150	4	3	Wine	178	13	3

4 Caso de estudio

Para mostrar las características del módulo educativo, se va a realizar un pequeño experimento involucrando dos algoritmos basados en sistemas difusos. El objetivo del estudio será destacar qué técnica ofrece mejores resultados en clasificación sobre un determinado conjunto de problemas de clasificación supervisada.

La Tabla 1 describe los 10 conjuntos seleccionados y sus principales características: **# Ej.** indica el número de ejemplos (instancias) del conjunto, **# At.** indica el número de atributos y **# Cl.** indica el número de clases. Estos conjuntos han sido tomados del repositorio KEEL-Dataset repository [3] [3], estando directamente disponibles en la instalación estándar del módulo educativo. Como esquema de validación, se ha seleccionado una validación cruzada en 10 partes (en general, el uso de este esquema es preferible a emplear una simple partición de entrenamiento y test, debido a que los resultados obtenidos suelen ser menos sensibles al sobreaprendizaje, y además permite que toda instancia disponible pertenezca al conjunto de test una vez).

Como algoritmos de Sistemas Difusos a estudiar, se ha seleccionado el algoritmo de Chi *et al.* [6], en sus versiones con 3 y 5 etiquetas por variable (**Chi-3** y **Chi-5**). Ambas versiones de este Sistema Difuso clásico serán comparadas con **SLAVE** [10], un competente Sistema Difuso Evolutivo. Puede encontrarse más información sobre ambos métodos tanto en la ayuda del propio módulo educativo como en el sitio web del proyecto KEEL, en la sección de algoritmos incluidos (*Included Algorithms*) [4]).

En KEEL, cada método ofrece una configuración de parámetros por defecto (configurada siguiendo las recomendaciones de sus autores). En este estudio, se emplearán dichos parámetros, con excepción del número de etiquetas para las 2 versiones del algoritmo de Chi *et al.*.

Una vez seleccionados los conjuntos de datos, algoritmos, parámetros y esquema de validación a emplear, el siguiente paso consiste en dibujar el grafo que representa al experimento (ver Figura 5). Una vez creado, el experimento puede comenzar. La interfaz de control (Figura 6) permitirá en este momento iniciar el experimento, así como pausarlo o terminarlo en caso de ser necesario.

Una vez finalizado el experimento, la interfaz mostrará los resultados para cada método y partición. Para cada uno, se muestra un informe completo incluyendo los resultados en acierto y las matrices de confusión obtenidas, tanto en entrenamiento como en test (ver Figura 7).

[3] `http://www.keel.es/datasets.php`
[4] `http://www.keel.es/algorithms.php`

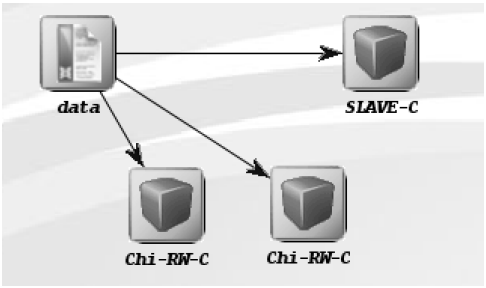

Figura 5: Grafo del experimento preparado para este caso de estudio

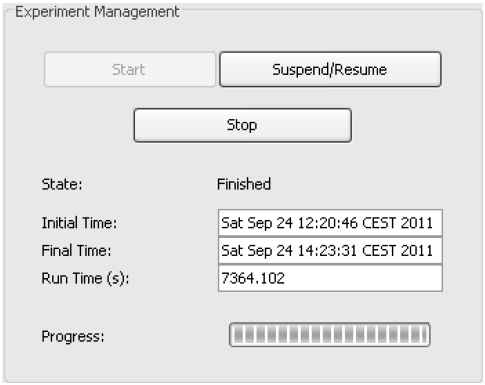

Figura 6: Interfaz de control para gestionar los experimentos del módulo educativo

```
Set:test
Total percentage of successes:
0.96
Percentage of successes in each partition:
1       0.933
2       0.933
3       1.0
4       1.0
5       1.0
6       0.933
7       0.933
8       0.933
9       1.0
10      0.933
Confusion matrix (rows=real class;columns=obtained class):
49      1       0
0       45      5
0       0       50
```

Figura 7: Resultados obtenidos por SLAVE en el problema Iris. El informe muestra el acierto obtenido en cada partición, el acierto medio y la matriz de confusión generada.

Chi-3

```
@Number of rules: 14

1: sepalLength IS L_0 AND sepalWidth IS L_1 AND petalLength IS L_0 AND petalWidth IS L_0: Iris-setosa with Rule Weight: 0.9946914141525106
2: sepalLength IS L_1 AND sepalWidth IS L_2 AND petalLength IS L_0 AND petalWidth IS L_0: Iris-setosa with Rule Weight: 1.0
3: sepalLength IS L_1 AND sepalWidth IS L_1 AND petalLength IS L_0 AND petalWidth IS L_0: Iris-setosa with Rule Weight: 0.9897080145841041
4: sepalLength IS L_0 AND sepalWidth IS L_0 AND petalLength IS L_0 AND petalWidth IS L_0: Iris-setosa with Rule Weight: 0.8743721633888051
5: sepalLength IS L_1 AND sepalWidth IS L_1 AND petalLength IS L_1 AND petalWidth IS L_1: Iris-versicolor with Rule Weight: 0.5474870380145459
6: sepalLength IS L_1 AND sepalWidth IS L_0 AND petalLength IS L_1 AND petalWidth IS L_1: Iris-versicolor with Rule Weight: 0.672414074237995
7: sepalLength IS L_0 AND sepalWidth IS L_0 AND petalLength IS L_1 AND petalWidth IS L_1: Iris-versicolor with Rule Weight: 0.7985932110408883
8: sepalLength IS L_1 AND sepalWidth IS L_1 AND petalLength IS L_2 AND petalWidth IS L_2: Iris-virginica with Rule Weight: 0.872291803923327
9: sepalLength IS L_2 AND sepalWidth IS L_1 AND petalLength IS L_2 AND petalWidth IS L_2: Iris-virginica with Rule Weight: 0.9575702462535564
10: sepalLength IS L_1 AND sepalWidth IS L_1 AND petalLength IS L_2 AND petalWidth IS L_1: Iris-virginica with Rule Weight: 0.18040318939382682
11: sepalLength IS L_2 AND sepalWidth IS L_1 AND petalLength IS L_2 AND petalWidth IS L_1: Iris-virginica with Rule Weight: 0.6509993328176151
12: sepalLength IS L_1 AND sepalWidth IS L_0 AND petalLength IS L_2 AND petalWidth IS L_1: Iris-virginica with Rule Weight: 0.1843982387098268
13: sepalLength IS L_1 AND sepalWidth IS L_1 AND petalLength IS L_1 AND petalWidth IS L_2: Iris-virginica with Rule Weight: 0.6473680231744567
14: sepalLength IS L_1 AND sepalWidth IS L_0 AND petalLength IS L_1 AND petalWidth IS L_2: Iris-virginica with Rule Weight: 0.6585359964748987
```

SLAVE

```
@Number of rules: 3

IF  petalWidth = ( L0 ) THEN Class IS Iris-setosa   W 1.0

IF  petalLength = ( L0 L1 L2 ) petalWidth = ( L0 L1 L2 ) THEN Class IS Iris-versicolor   W 0.4789917154426708

IF  petalWidth = ( L0 L1 L3 L4 ) THEN Class IS Iris-virginica   W 0.4123616236162361
```

Figura 8: Bases de reglas generadas por **Chi-3** (arriba) y **SLAVE** (abajo) en Iris. En ambos casos puede comprobarse la influencia de la longitud y anchura del pétalo para clasificar, si bien la base de reglas de SLAVE es mucho más compacta.

Tabla 2: Resultados obtenidos (Acierto en test)

Conjunto	**Chi-3**	**Chi-5**	**SLAVE**
Bupa	0.5790	**0.5940**	0.5850
Ecoli	0.7200	0.8180	**0.8510**
Glass	**0.5980**	0.5930	0.5790
Haberman	**0.7320**	0.7250	0.7120
Iris	0.9260	0.9530	**0.9600**
Monk-2	0.4280	0.4720	**0.9740**
New-thyroid	0.8410	**0.9110**	**0.9110**
Pima	**0.7300**	0.7170	0.7260
Vehicle	0.6070	**0.6400**	0.6180
Wine	**0.9380**	0.7520	0.9320
Average	0.7099	0.7175	**0.7848**

La Tabla 2 muestra los resultados medios obtenidos en acierto (expresados en el intervalo $[0,1]$). Los mejores resultados están destacados en **negrita**. Como puede verse, no existe un método que domine claramente la comparación en todos los casos, si bien SLAVE obtiene un acierto medio mayor. Este es un resultado interesante, ya que sugiere que los problemas seleccionados para el experimento poseen diferentes cualidades que los hacen más sencillos o complejos para cada algoritmo. Así, puede ser interesante tratar de analizar los motivos de estas diferencias.

Otro aspecto interesante consiste en analizar los modelos generados. El módulo educativo ofrece, junto a los resultados obtenidos, un listado de los modelos generados durante el entrenamiento. Esto es importante de cara a determinar cómo está funcionando un determinado algoritmo y por qué motivo está ofreciendo resultados más o menos comunes.

En el caso de los Sistemas Difusos basados en reglas, es posible obtener la base de conocimiento generada durante el entrenamiento. De esta manera, los estudiantes pueden tratar de interpretar el conocimiento extraído por el algoritmo, valorando los modelos generados y comparándolos entre sí. Esta tarea, íntimamente ligada a como se analizarían los algoritmos en un estudio más complejo, puede ser realizada fácilmente gracias al módulo educativo.

Por ejemplo, la Figura 8 muestra las bases de reglas generadas por **Chi-3** y **SLAVE** para el problema Iris. De éste problema se conoce que los atributos relativos a la longitud y anchura del pétalo son los más determinantes a la hora de establecer una clasificación. En efecto, si estudiamos la base de reglas obtenida por **Chi-3** podemos ver cómo los ejemplos de la clase *Iris-setosa* quedan caracterizados por aquellas reglas que asignan la etiqueta L_0 a los dos atributos del pétalo (*petalLength y petalWidth*). Los ejemplos de la clase *Iris-versicolor* se caracterizan por tener asignada la etiqueta L_1 a ambos atributos, mientras que los ejemplos de la clase *Iris-virginica* deben de tener asignada la etiqueta L_2 en al menos uno de estos 2 atributos clave.

La base de reglas de **SLAVE** también usa los atributos del pétalo, pero de forma más simple: La clase *Iris-setosa* queda caracterizada por los ejemplos cuya anchura de pétalo tenga asignada la etiqueta L_0. Si ambas medidas de pétalo tienen asignadas las etiquetas L_0, L_1 o L_2, la clase de la instancia será *Iris-versicolor* (con menor prioridad que los ejemplos de *Iris-setosa*). Finalmente, aquellos ejemplos cuya anchura de pétalo no sea L_2 y no hayan sido clasificados por las reglas anteriores, serán considerados como instancias de la clase *Iris-virginica*.

Ambos ejemplos contrastan en este caso, sobre todo si se tiene en cuenta que **SLAVE** ofrece un mayor acierto en test. En este caso, un alumno podría concluir que el motivo de la mejor precisión ofrecida por **SLAVE** se debe a la mayor generalidad de sus reglas (de 1 o 2 condiciones), lo cual permite clasificar de forma más consistente nuevos ejemplos no considerados durante el entrenamiento.

En resumen, este caso de estudio ha mostrado las principales características del módulo educativo de KEEL como apoyo a la docencia de sistemas difusos. El módulo permite a los estudiantes poder emplear y analizar métodos de diferentes áreas sin necesidad de implementarlos. Así, pueden dedicar su esfuerzo a analizar los resultados obtenidos y así tratar de encontrar los motivos de las diferencias entre métodos y/o la forma de optimizar su comportamiento.

5 Conclusiones

En este trabajo hemos presentado KEEL como herramienta de apoyo a la docencia de Sistemas Difusos. Mediante su módulo educativo, es posible acceder de forma sencilla a varios problemas y técnicas del estado del arte, permitiendo así su uso por alumnos y profesores en asignaturas y cursos introductorios al área.

Se ha desarrollado un caso de uso con el objetivo de demostrar las capacidades del módulo. Este caso de uso ha consistido en una comparación entre varios Sistemas Difusos sobre 10 problemas de clasificación. Tras su realización, se han analizado los resultados obtenidos a través de la herramienta, así como los modelos obtenidos. Esto ha permitido ilustrar cómo puede obtenerse información relevante acerca del comportamiento de un método mediante las herramientas disponibles en el módulo.

Gracias a ello, han podido extraerse varias conclusiones de forma sencilla, sin necesidad de implementar ninguno de los Sistemas Difusos considerados ni de realizar ningún esfuerzo de programación adicional para recuperar y analizar los resultados. De esta manera, pueden realizarse experimentos similares en la docencia de éstas u otras técnicas incluidas en el módulo, de forma asequible para los profesores o alumnos que lo necesiten.

Agradecimientos

Este trabajo ha sido soportado por los proyectos nacionales TIN2008-06681-C06-01 y TIN2011-28488, y por el proyecto andaluz TIC-2010-6858. J. Derrac posee una beca FPU del Ministerio de Educación.

Referencias

[1] T. Abeel, Y. V. de Peer, Y. Saeys: Java-ML: A machine learning library. *Journal of Machine Learning Research* 10, pp. 931–934, 2009.

[2] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, F. Herrera: KEEL: A software tool to assess evolutionary algorithms to data mining problems. *Soft Computing* 13:3, pp. 307–318, 2009.

[3] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera: Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing* 17:2–3, pp. 255–287, 2011.

[4] G. E. A. P. A. Batista, R. C. Prati, M. C. Monard: A study of the behaviour of several methods for balancing machine learning training data. *SIGKDD Explorations* 6:1, pp. 20–29, 2004.

[5] R. R. Bouckaert, E. Frank, M. A. Hall, G. Holmes, B. Pfahringer, P. Reutemann, I. H.Witten: Weka - experiences with a java open-source project. *Journal of Machine Learning Research* 11, pp. 2533–2541, 2010.

[6] Z. Chi, H. Yan, T. Pham: *Fuzzy algorithms with applications to image processing and pattern recognition* World Scientific, Nueva York, 1996.

[7] J. Derrac, S. García, D. Molina, F. Herrera: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* 1:1, pp. 3–18, 2011.

[8] T. Dietterich, R. Lathrop, T. Lozano-Pérez: Solving the multiple instance problem with axis-parallel rectangles. *Artifical Intelligence* 89:1–2, pp. 31–71, 1997.

[9] S. García, F. Herrera: An extension on S̈tatistical comparisons of classifiers over multiple data setsS̈for all pairwise comparisons. *Journal of Machine Learning Research* 9, pp. 2579–2596, 2008.

[10] A. González, R. Pérez: Selection of relevant features in a fuzzy genetic learning algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 31:3, pp. 417–425, 2001.

[11] E. Mjolsness, D. DeCoste: Machine learning for science: State of the art and future prospects. *Science* 293, pp. 2051–2055, 2001.

[12] S. Sonnenburg, M. Braun, C. Ong, S. Bengio, L. Bottou, G. Holmes, Y. LeCun, K.-R. Müller, F. Pereira, C. Rasmussen, G. Rätsch, B. Schölkopf, A. Smola, P. Vincent, J. Weston, R. Williamson: The need for open source software in machine learning. *Journal of Machine Learning Research* 8, pp. 2443–2466, 2007.