Actas XVIII JENUI 2012, Ciudad Real, 10-13 de julio 2012
I.S.B.N. 10: 84-615-7157-6 | I.S.B.N. 13:978-84-615-7157-4
Páginas 33-40

Herramienta de apoyo a la docencia de algoritmos de selección de instancias

Álvar Arnáiz González,José Francisco Diez Pastor,César García Osorio,Juan José Rodríguez Díez
Universidad de Burgos
Avenida Cantabria s/n, 09006 Burgos
{alvarag,jfdpastor,cgosorio,jjrodriguez}@ubu.es

Resumen

En el currículo de ingeniería informática la minería de datos y el aprendizaje automático son cada vez más relevantes, tanto en los cursos de grado y máster, como también en los de doctorado. Prueba de ello es la aparición de diversas herramientas que facilitan el aprendizaje de algoritmos relacionados con la disciplina, mediante la ejecución paso a paso de los mismos y la visualización de los resultados. Sin embargo, para el caso concreto de los algoritmos de selección de instancias, estas herramientas son prácticamente inexistentes.

En el presente recurso docente se presenta una herramienta implementada para cubrir esta carencia. «Instance Selection», que es como se llama la aplicación, está preparada para mostrar el funcionamiento tanto de los algoritmos clásicos como alguno de los más modernos, permitiendo la ejecución paso a paso y visualizando los resultados intermedios para facilitar la labor didáctica.

Las principales ventajas de la aplicación descrita en este recurso docente son: que implementa varios algoritmos, lo que permite su comparación, es multiplataforma, permite la visualización incremental de los pasos de los algoritmos implementados, la interfaz está preparada para varios idiomas e incluye una completa ayuda.

Summary

In computer engineering curricula Data Mining and Machine Learning are increasingly important in both undergraduate and masters, as well as the PhD courses. The emergence of several tools that facilitate learning algorithms related to the discipline proves that. Some of these tools allow the execution of algorithms step by step showing the results of each step, others let the student change the algorithm parameters and the student can visualize the results. However, for the specific case of instance selection algorithms these tools are virtually nonexistent.

This paper discusses a tool implemented to fill this gap. “Instance Selection”, which is the name of the application, is prepared to show the operation of both classical instance selection algorithms as some of the most modern, allowing the execution step by step and displaying the intermediate results to facilitate the teaching task.

The main advantages of the application described in this teaching resource are that it implements several algorithms, allowing comparison between them, it is multi-platform, it allows the interactive visualization of the steps of the implemented algorithms, the interface is ready for several languages, it includes comprehensive help.

Palabras clave

Minería de datos, Selección de Instancias, Visualización de Algoritmos

1. Introducción

Cuando uno piensa en una técnica de minería de datos se imagina la aplicación de un algoritmo como la construcción de un árbol de decisión a partir de un conjunto de datos (en este artículo por conjunto de datos entendemos un conjunto de instancias en las que cada una de éstas es un conjunto de valores para los atributos que la caracterizan). Pero un paso previo y muy relevante de los procesos de minería de datos es la etapa de pre-procesamiento del conjunto de datos para adecuarlo al algoritmo de minería de datos que se encargará de extraer conocimiento del mismo. La importancia de esta etapa radica en el profundo impacto que tiene en el éxito de los pasos posteriores de extracción de conocimiento. Sin entrar en los detalles de cada una de ellas, las tareas de pre-procesamiento comprenden entre otras las siguientes:

- Integración de datos y enlazado: combinación de varias fuentes de datos en un único conjunto de datos.
- Limpieza de datos: eliminación de valores alterados debidos a ruido y datos atípicos, tratamiento de valores ausentes, corrección de inconsistencias.
- Transformación de datos: normalización de valores, agregación y resumen de instancias, construcción de atributos derivados.
- Reducción de datos y discretización: los valores de los atributos se pueden discretizar, algunos de los atributos se pueden descartar, el tamaño del conjunto de datos se puede reducir mediante el muestreo y la selección de instancias.

En este recurso se presenta una herramienta que facilita el aprendizaje de las técnicas de selección de instancias.

Tras esta introducción, el resto del artículo se estructura como sigue: en la siguiente sección se proporciona un breve contexto teórico; la sección 3 lista algunas otras herramientas y enumera las ventajas que la propuesta tiene sobre aquellas; la sección 4 lista los algoritmos implementados por la aplicación; en la sección 5 se detalla la funcionalidad ofrecida por la herramienta; sobre la experiencia de su uso y sobre su disponibilidad se habla en la sección 6; por último, en la sección 7 se presentan las conclusiones y el trabajo futuro.

2. Contexto teórico

En la vida real, los conjuntos o bases de datos resultan extremadamente grandes por lo que muchos algoritmos tienen dificultades relacionadas con el tiempo de ejecución y el almacenamiento. Por todo ello, procesar la información disponible puede llegar a convertirse en una tarea difícil y problemática. Cuando se desea utilizar algún algoritmo de aprendizaje basado en instancias, como puede ser la regla del vecino más cercano, este problema puede ser determinante e impedir su ejecución.

Para evitar este problema existen técnicas que persiguen reducir el tamaño de los conjuntos de datos como la selección de instancias. En la selección de instancias se agrupan toda una serie de procedimientos y algoritmos que tienen como finalidad la selección de un subconjunto representativo del conjunto de entrenamiento inicial. El objetivo general es intentar eliminar del conjunto de entrenamiento aquellas instancias erróneamente clasificadas y, al mismo tiempo, reducir los posibles solapamientos entre regiones de clases distintas, es decir, su objetivo principal es lograr agrupamientos compactos y homogéneos. De forma más concreta se puede pensar que se persigue seleccionar un subconjunto de instancias que proporcione un comportamiento para la *Regla Nearest Neighbor (Vecino más cercano o kNN)* similar al obtenido utilizando la totalidad del conjunto de entrenamiento.

Tradicionalmente se han clasificado los métodos de selección de instancias, en función de la dirección de la búsqueda, siguiendo varias estrategias: incremental, decremental, por lotes o mixta. A continuación se detalla cada una de ellas.

- Estrategia incremental: se parte de un conjunto vacío S de instancias seleccionadas, en cada paso se añade a S la instancia que satisface el criterio empleado para la selección de instancias. En esta estrategia, el orden en que se presentan las instancias en el conjunto es importante, ya que la probabilidad de que las primeras instancias sean incluidas en S es mayor que la de las últimas. En este sentido, puede verse dañada la precisión en la clasificación si las últimas instancias representan una mayor generalización que las primeras. La ventaja de este tipo de estrategia es que suele ser más rápida y consume menos recursos de almacenamiento.
- Estrategia decremental: comienza considerando el conjunto de instancias seleccionadas igual al conjunto de instancias de partida, $S = T$, y en cada paso, se determina la instancia a eliminar de S de acuerdo al criterio de selección de instancias. Al igual que en la anterior, en esta estrategia es importante el orden en que las instancias se presentan pero, a diferencia de las técnicas incrementales, todas las instancias parcialmente almacenadas están disponibles en todo momento para examinar cuál de ellas resulta conveniente eliminar. La principal ventaja de estas estrategias es que obtienen una mayor reducción del conjunto de entrenamiento y, normalmente, logran una mayor precisión en la clasificación. Por otra parte, estas estrategias suelen ser computacionalmente más costosas.

- Estrategia por lotes: consiste en identificar y marcar en cada paso aquellas instancias que no satisfacen el criterio de selección, las cuales no serán consideradas en el subconjunto *S* y, finalmente, eliminarlas. En cada paso, no se elimina sólo una instancia sino grupos de éstas. Al igual que la estrategia decremental, esta técnica resulta ser costosa desde el punto de vista computacional.
- Estrategia mixta: combina pasos incrementales y decrementales de forma alterna.

3. Otras herramientas

En esta sección se van a repasar algunas de las bibliotecas y aplicaciones relacionadas con la selección de instancias que pueden encontrarse en la red.

3.1. Bibliotecas

A continuación se repasan un par de bibliotecas con algoritmos de minería de datos. Aunque ninguna de ellas tiene como objetivo facilitar el aprendizaje de algoritmos como la aplicación descrita en este recurso docente. La primera de ellas es más completa y general mientras que la siguiente está totalmente enfocada a la selección de instancias.

- KEEL [12] es una completa biblioteca de algoritmos de minería de datos escrita en Java. Ofrece soluciones para problemas de regresión, clasificación, clustering... Implementa multitud de algoritmos, entre los cuales se encuentran algunos de selección de instancias, el problema es que no puede ser utilizada directamente para el aprendizaje de los mismos ya que no permite visualizar su funcionamiento.
- La biblioteca de Nicolás García-Pedrajas [13] implementa los algoritmos de selección de instancias más comunes. Consiste en una biblioteca escrita en C que, al igual que la anterior, solo permite ejecutar los algoritmos, y no ver su funcionamiento.

3.2. Aplicaciones gráficas

Las aplicaciones gráficas que se pueden encontrar se centran en un algoritmo concreto, no permitiendo comparar resultados con otras técnicas de selección de instancias. A continuación se citan algunos «applets» que pueden ser utilizados para la docencia.

- En [10] puede visualizarse el funcionamiento de un clasificador basado en el algoritmo del vecino más próximo (Nearest Neighbor). El *canvas* dibuja aleatoriamente un número de puntos (instancias) que servirán para clasificar las instancias que introduzca el usuario.
- En esta página de la Universidad de Leicester [8] se puede dibujar un conjunto de instancias en dos dimensiones y aplicar el algoritmo CNN (*Condensed Nearest Neighbor*). El algoritmo es ejecutado completamente por lo que el usuario solo puede consultar el resultado final.
- La *Iterative Case Filtering Demo* [9] permite visualizar el funcionamiento del algoritmo ICF. Lo atractivo del mismo es que muestra la ejecución paso a paso, lo cual ayuda a comprender el funcionamiento del mismo. Es similar en filosofía al recurso presentado en este artículo, pero más limitada, en el sentido que sólo muestra el funcionamiento de un algoritmo.
- Como en el caso anterior, esta página [11] muestra el funcionamiento del algoritmo Híbrido de Gabriel Graph. La aplicación permite dibujar instancias de dos clases y ejecutar los tres pasos del algoritmo, mostrando la triangulación de Delaunay.

Frente a todas las anteriores, las principales ventajas que presenta este recurso docente vienen dadas por el número de algoritmos soportados y la capacidad de visualizar los pasos de los métodos implementados. También cabe destacar que la herramienta es multiplataforma, que su interfaz está preparada tanto para inglés como para castellano y que incluye una completa ayuda.

4. Algoritmos implementados

La aplicación presentada en este artículo implementa los algoritmos más representativos de la disciplina. Éstos se han seleccionado para cubrir un amplio abanico de métodos de selección de instancias.

- Algoritmo Condensado de Hart (CNN *Condensed Nearest Neighbor*) [1]: se considera la primera propuesta formal de condensado

para la regla NN. Un elemento teórico importante presente en este algoritmo es el concepto de consistencia respecto al conjunto de entrenamiento, que se define como sigue: Sea $X \neq \emptyset$ un conjunto y consideremos $S \subseteq X$, decimos que el subconjunto S es consistente respecto al conjunto X si, al utilizar al subconjunto S como conjunto de aprendizaje, se puede clasificar correctamente a todo el conjunto X.

- Algoritmo Condensado Reducido (RNN): Gates [3] realiza una extensión decremental de CNN con la regla del vecino más cercano reducido (RNN), la cual comienza con $S = SCNN$, donde $SCNN$ es el conjunto de instancias obtenido con CNN a partir del conjunto de partida T. Posteriormente, se elimina una instancia de S si dicha eliminación no causa que alguna otra instancia de T sea clasificada erróneamente al considerar la vecindad a las instancias restantes de S. Con esta regla se obtiene un subconjunto de CNN, por lo que el conjunto editado, en cuanto al número de instancias, es menor que el obtenido con CNN.
- Algoritmo Subconjunto Selectivo Modificado (MSS): sin querer ahondar mucho, en el subconjunto selectivo [4]; se puede ver como la idea del algoritmo de condensado de Hart, pero introduciendo una condición más fuerte que la condición de consistencia, con el objetivo de buscar aquellas instancias en un orden independiente y de una manera más conveniente. Para satisfacer este objetivo introduce la definición de subconjunto selectivo (SS). Una propuesta de modificación al algoritmo del Subconjunto Selectivo es la del algoritmo Subconjunto Selectivo Modificado [5]. Los autores introducen una modificación en la definición de subconjunto selectivo, utilizan las definiciones de vecino relacionado y de vecindad relativa propuestas por Ritter, pero sustituyendo la definición de subconjunto selectivo mínimo.
- Algoritmos Decremental Reduction Optimization Procedure (DROP): estos métodos decrementales [6] basan su regla de selección en términos del concepto de socio y de asociado que se definen como sigue:

  - Socio: sea $X \neq \emptyset$, el socio de una instancia P que pertenece al conjunto X, es aquella instancia que tiene a P como uno de sus k vecinos más cercanos.
  - Asociado: aquellas instancias que tienen a P como uno sus k vecinos más cercanos son llamadas asociados de P y se denotan mediante la expresión $P.A_{1,\ldots,a}$, donde a es el número de asociados de P.

- Algoritmo Iterative Case Filtering (ICF): La regla de selección del ICF [9] se basa en los conceptos de *alcance* y *cobertura* que corresponden a *vecindario* y *conjunto de socios* respectivamente. La regla de selección consiste en eliminar todas aquellas instancias tales que la cardinalidad del conjunto *alcance* sea superior al de *cobertura*. Esto quiere decir que se eliminará una instancia cuando existe otra intancia que generaliza la información que pudiese proporcionar. Para evitar el efecto nocivo del ruido ICF comienza con un filtrado por Wilson [2].
- Algoritmo Democratic Instance Selection (DIS): el algoritmo DIS [7] consiste en hacer un número de rondas r para cada algoritmo de selección de instancias aplicado a un número de particiones sobre el conjunto original. Para cada ronda el proceso consiste en dividir el conjunto original en un número de subconjuntos disjuntos (de aproximadamente el mismo tamaño). Sobre cada subconjunto se aplica un algoritmo de selección de instancias clásico de manera separada. Las instancias seleccionadas por el algoritmo para ser eliminadas recibirán un voto. Entonces una nueva ronda comienza y el proceso se repite. Después de un número definido de rondas, las instancias que están por encima de un valor umbral son borradas.

Como puede verse, el conjunto de soluciones a la selección de instancias es amplio y variado. La descripción que se ha dado aquí ha sido necesariamente breve por la imposición de las limitaciones de espacio, pero la compresión de los conceptos utilizados en estos algoritmos tampoco es inmediata tras una lectura inicial de los detalles del método. Una herramienta como la presentada en este recurso docente facilita y acelera el aprendizaje de las particularidades de cada método.

5. Utilizando la herramienta

La interfaz gráfica de la aplicación se caracteriza por la simplicidad: un menú básico de herramientas, el panel lateral izquierdo para la introducción de datos y el área central para la ejecución de los algoritmos. En el lateral izquierdo se distinguen dos secciones:

- Conjunto de datos: es aquí donde el alumno escoge el conjunto de datos (*dataset*). Una vez cargado se seleccionan los atributos clase y los que se representarán en cada uno de los ejes (de este modo se elige que dos dimensiones utilizar para visualizar el conjunto de datos cuando éste tiene más de dos dimensiones) y el porcentaje de instancias que serán utilizadas para la comprobación. Estas instancias no serán utilizadas para la selección.
- Parámetros del algoritmo: en función del algoritmo que se escoja en la *combo-box* se habilitarán una serie de campos, como pueden ser el número de vecinos a tener en cuenta.

Una vez seleccionado el conjunto de datos y escogido el algoritmo, pulsando sobre el botón «*Comenzar*», la aplicación preguntará al usuario si desea preprocesar los datos tal y como se ve en la Figura 1. La normalización, estandarización o tipificación sirve para evitar que atributos con mucha varianza «*monopolicen*» la «vecindad» de las instancias.

Tras el preprocesado, se abre una nueva pestaña (Figura 2) donde tiene lugar la ejecución del algoritmo. La pestaña se encuentra dividida en tres secciones:

- Canvas: la parte central la ocupa una gráfica en dos dimensiones donde se representa el conjunto de instancias sobre el que se van a aplicar los algoritmos. Las instancias seleccionadas por el algoritmo son representadas con un punto grueso mientras que las rechazadas con un aspa. El color de la instancia es lo que determina la clase a la que pertenece. Pulsando con el ratón sobre cualquiera de las instancias aparece una ventana emergente con el valor de cada uno de los atributos de la instancia.
- Panel lateral derecho: en su parte superior aparece el pseudocódigo del algoritmo, el cual se irá iluminando a medida que se ejecute cada uno de los pasos del mismo. Debajo, un cuadro de texto muestra el progreso, indicando la inclusión o rechazo de cada una de las instancias.
- Botonera: la parte inferior de la pestaña es la que se encuentra reservada a la interacción con el usuario, quien pulsando repetidamente sobre el botón «*Avanzar paso*» ejecutará el algoritmo a su voluntad. También permite una ejecución continua y avanzar hasta el final ejecutando todos los pasos que sean necesarios. Dos «checkbox» sirven para mostrar u ocultar las regiones de Voronoi y todas las instancias del conjunto de entrenamiento.

Una vez se han ejecutado todos los pasos del algoritmo, se habilita el botón «*Evaluar algoritmo*», el cual al ser pulsado crea la pestaña de evaluación. En dicha pestaña se muestra el conjunto de datos seleccionado por el algoritmo así como una breve estadística de su ejecución: tiempo empleado, número de instancias del conjunto inicial, instancias seleccionadas y el porcentaje de error del clasificador entrenado con las instancias seleccionadas por el algoritmo.

Las regiones de Voronoi permiten al alumno conocer la influencia o vecindad de cada una de las clases de instancias. De un modo visual se aprecia con facilidad si la instancia tiene a su vecino más próximo de la misma clase o de otra distinta.

Gracias a la ejecución paso a paso, el alumno puede visualizar qué instancia se está evaluando en cada momento (es marcada con un rombo) y qué fragmento del algoritmo. De este modo puede seguir el pseudocódigo y entender la decisión de escoger o descartar la instancia analizada.

Otro de los puntos fuertes de la herramienta es la ayuda en línea. Esto permite a los usuarios poder acceder al soporte sin necesidad de documentos externos. De este modo, cualquier duda que le pueda surgir al alumno durante la ejecución puede resolverla con tan sólo pulsar *F1*.

6. Experiencia y disponibilidad

La utilidad de la herramienta como recurso docente queda desglosada en secciones anteriores, al estar disponible bajo licencia GNU/GPL la hace totalmente accesible a la comunidad docente.

Como se ha indicado, la accesibilidad aumenta al estar traducida a cuatro idiomas: inglés, alemán,

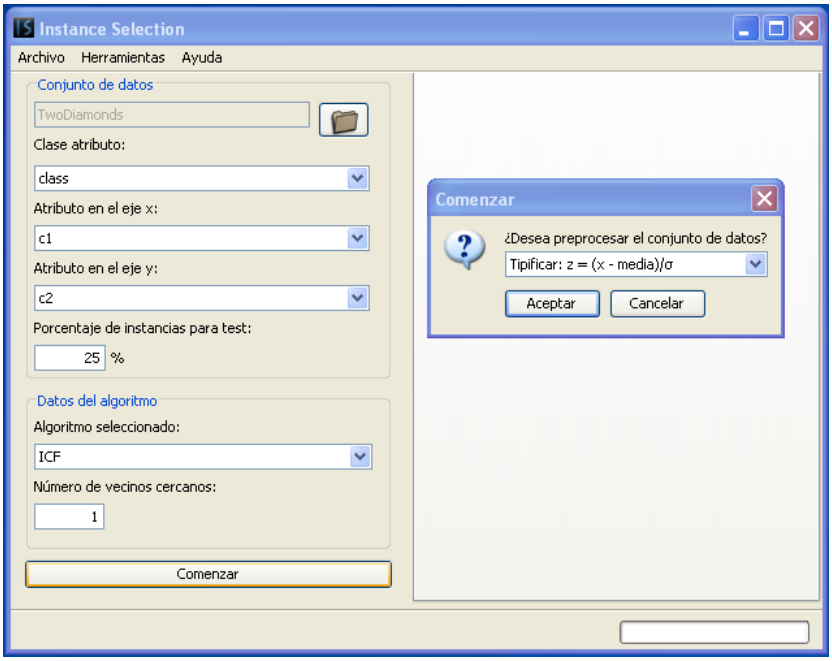

Figura 1: Comenzando la ejecución de un algoritmo

francés y castellano. De las herramientas mencionadas en apartados anteriores, ninguna de ellas se encontraba en castellano, siendo esto un motivo que puede dificultar el aprendizaje de los algoritmos en algunos países.

Actualmente existe una página web (`http://pisuerga.inf.ubu.es/cgosorio/ISBur`) desde la cual puede descargarse la aplicación. Como se ha programado en Java es portable a cualquier plataforma y carece de instalación, lo único necesario es copiar el «*.jar*» y la carpeta de librerías en el equipo donde se desea utilizar.

Debido a la novedad de la herramienta, todavía no ha podido ser utilizada en los grados de informática ni en los másteres que se imparten en la Universidad de Burgos. Este es el motivo por el cual la experiencia que se detalla es la de los alumnos involucrados en el desarrollo y no en usuarios finales.

Aunque la teoría de selección de instancias puede ser reducida en unas cuantas diapositivas, gracias a lo compacta que se puede hacer la especificación de sus algoritmos, la realidad es que cada uno de ellos alberga una gran complejidad. Introducen nuevos conceptos o redefinen alguno ya conocido, esto complica la comprensión de los mismos y puede desanimar al alumnado.

Los alumnos que intervinieron en el desarrollo de la herramienta valoraron muy positivamente los diversos «*applets*» que de una manera visual explicaban algunos algoritmos. Esto es lo que se ha buscado en el desarrollo de la aplicación desde el inicio, ayudando la comprensión a través del avance paso a paso y de la iluminación del fragmento de pseudocódigo que se está ejecutando en un momento concreto.

7. Conclusiones y trabajo futuro

Tras analizar la necesidad del mismo, en este artículo se ha presentado un recurso docente que facilita el aprendizaje de los métodos de selección de instancias mediante técnicas de visualización de algoritmos. En concreto, este recurso permite la ejecución paso a paso de los algoritmos implementados de modo que el estudiante pueda ver el efecto que tiene cada uno de ellos en el conjunto de datos sobre el que se aplica.

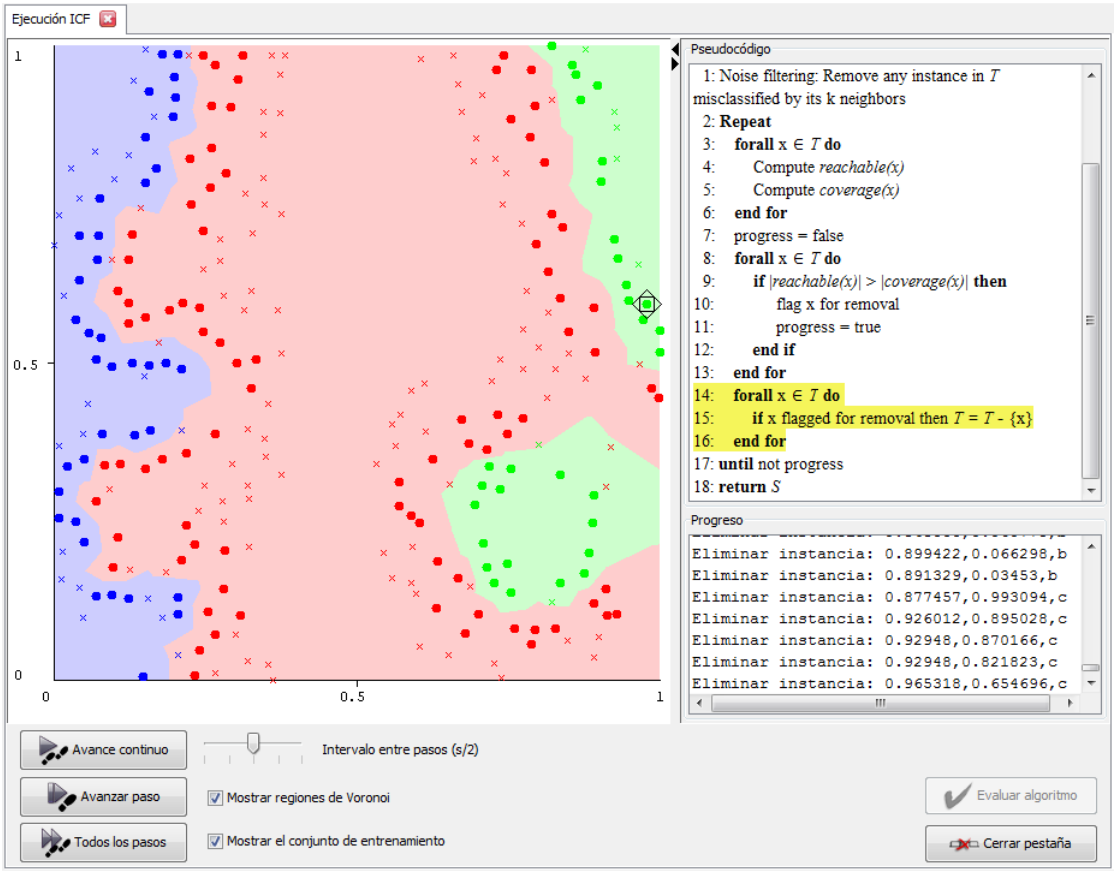

Figura 2: Ejecutando el algoritmo ICF en la aplicación

La aplicación tiene una completa ayuda que hace que ésta sea autocontenida, además ésta está indexada para facilitar la búsqueda.

La herramienta es capaz de obtener conjuntos de datos de diversos orígenes (*ARFF, CSV* y bases de datos). Esta característica presta versatilidad a la hora de analizar conjuntos de instancias.

La aplicación aquí presentada se ha preparado para que pueda evolucionar y mejorar en el futuro. Durante la fase de diseño se ha tenido muy en cuenta la realización de un núcleo robusto y extensible que permita posteriores ampliaciones y mejoras. Por ello, los algoritmos han sido diseñados para que acceder a la información de sus estructuras internas sea fácil. Esto posibilitará futuras mejoras que permitan un mayor nivel de granularidad en la ejecución paso a paso.

Se espera poder completar la página web a través de la cual se distribuye con información adicional relevante al área de estudio de la selección de instancias. Además, en la versión actual la ayuda sólo está disponible en castellano, en futuras versiones se planea incluir también en los otros idiomas para los que está preparada la interfaz de la aplicación.

Referencias

[1] Hart *The Condensed Nearest Neighbor Rule*, IEEE Transactions on Information Theory, 1968

[2] Wilson, *Asymptotic properties of nearest neighbour rules using edited data*, IEEE Transactions on Systems, 1972

[3] Gates, G.W. *The reduced nearest neighbour rule*, IEEE Transactions on Information Theory IT-18, 1972

[4] Ritter, G.L., Woodruff, H.B., Lowry, S.R. and Isenhur, T.L. *An algorithm for selective nearest neighbour decision rule* IEEE Transactions on Information Theory, 1975

[5] Ricardo Barandela and Francesc J. Ferri and J. Salvador Sánchez, *Decision boundary preserving prototype selection for nearest neighbor classification*, International Journal of Pattern Recognition and Artificial Intelligence, 2005

[6] D. Randall Wilson and Tony R. Martinez, *Reduction Techniques for Instance-Based Learning Algorithms*, Machine Learning, 2000

[7] García-Osorio, C et al., *Democratic Instance Selection: A linear complexity instance selection algorithm based on classifier ensemble concepts*, Artificial Intelligence, 2010

[8] University of Leicester *KNN and Potential Energy* `http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html` (última visita el 18 de abril de 2012)

[9] Edith L.M. Law *Instance Based Learning Tutorial* `http://cgm.cs.mcgill.ca/~athens/cs644/Projects/2004/SumedhaAhuja-EdithLaw/applet.html` (última visita el 18 de abril de 2012)

[10] Nascimento Ferreira, Leonardo *K-Nearest Neighbor (KNN) Classifier* `http://www.leonardonascimento.com/en/knn.html` (última visita el 18 de abril de 2012)

[11] McGill University `http://www.cs.mcgill.ca/~cs644/Godfried/2005/Fall/Kietzmann/en/applet.html` (última visita el 18 de abril de 2012)

[12] KEEL (Knowledge Extraction based on Evolutionary Learning) `http://www.keel.es/` (última visita el 18 de abril de 2012)

[13] García-Pedrajas, Nicolás *Universidad de Córdoba* `http://cibrg.org/documents/InsSel-7Jul2010.tgz` (última visita el 18 de abril de 2012)