Selección de atributos relevantes basada en bootstrapping

Jesús S. Aguilar–Ruiz y Norberto Díaz–Díaz
Dept.de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
ETS Ingeniería Informática
41012 Sevilla
{aguilar, ndiaz}@lsi.us.es

Resumen

Los resultados de los métodos de selección de atributos influyen en gran medida en el éxito de un proceso de minería de datos. Para asegurar el resultado óptimo en un método de selección se debe realizar una búsqueda exhaustiva, lo que conlleva un alto coste computacional que en ocasiones hace que sea inabordable. En este documento proponemos un método de selección de atributos basado en las técnicas de bootstrapping que reduce el coste de la búsqueda exhaustiva, sin reducir la precisión en la clasificación.

Los experimentos han sido realizados con bases de datos de alta dimensionalidad (miles de atributos) y muestran un rendimiento satisfactorio de la propuesta.

1 Introducción

El éxito de aplicar un algoritmo de minería de datos está sujeto a diferentes factores. La calidad de los datos de entrada, por ejemplo, es uno de estos factores, ya que si éstos contienen información irrelevante o redundante, o posee datos ruidosos, el procesos de aprendizaje será más dificil.

Los métodos de selección de atributos permiten identificar y eliminar parte de la información irrelevante y redundante. Este tipo de procesos consiste en seleccionar un subconjunto de atributos óptimo, a partir de los datos de entrada, que maximice la eficiencia de los algoritmos de minería de datos sobre dicha información.

Un proceso de selección de atributos se divide en cuatro etapas [1]: en la primera se determina el posible conjunto de atributos para realizar la representación del problema; en la segunda se evalúa el subconjunto de atributos generado en el paso anterior; el chequeo de si el subconjunto seleccionado satisface el criterio de detección de búsqueda se realiza en la tercera etapa; y por último, en la etapa cuarta se verifica la calidad del subconjunto de atributos que se determinó. Estos procesos se pueden clasificar de forma diferente dependiendo de la etapa en la que nos centremos para realizar dicha distinción. Por ejemplo, atendiendo a la función de evaluación usada (paso 2), los procedimientos de selección de atributos se pueden clasificar en tres categorías categorías [2]: Filters, Wrappers [7, 8] e híbridos [9].

En los métodos Filters el procedimiento de selección es realizado independientemente de la función de evaluación (clasificación). En éstos se pueden distinguir cuatros medidas de evaluación diferentes: distancia, información, dependencia y consistencia. Ejemplos respectivos de cada una de estas medidas son: ReliefS [3], DTM [4], POE&AAC [5] y SOAP [6].

Los métodos Wrappers combinan la búsqueda en el espacio de atributos con el algoritmo de aprendizaje, evaluando los conjuntos de atributos y escogiendo el más adecuado. El inconveniente que presentan es que son más costosos que los Filters [7] aunque suelen obtener mejores resultados.

Actas del III Taller Nacional de Minería de Datos y Aprendizaje, TAMIDA2005, pp.21-30
ISBN: 84-9732-449-8

Los modelos híbridos toman las ventajas que aportan los modelos Filters y Wrappers, aprovechando sus diferentes criterios de evaluación en diferentes etapas de búsqueda.

Independientemente de la función de evalucación elegida, los métodos de selección de atributos deben realizar una búsqueda entre los distintos candidatos de subconjuntos de atributos, pudiendo ser [2]: completa [10], secuencial [11] o aleatoria. La búsqueda completa (o exhaustiva) garantiza encontrar el resultado óptimo acorde al criterio de evaluación usado. El gran inconveniente es su elevado coste computacional ($\Theta(2^n)$) que hace que sea inabordable en caso de que el número de atributos (n) sea considerable. Las búsquedas secuenciales con un coste de $\Theta(n^2)$ no son completas, con lo que puede que no encontren el subconjunto óptimo. La búsqueda aleatoria, que tampoco asegura el óptimo, comienza con un subconjunto de atributos seleccionados aleatoriamente y procede, a partir de éste, con una busqueda secuencial (i.e. Random-Start [10]) o generando aleatoriamente el resto de subconjuntos candidatos (i.e. Algoritmo Las Vegas [12]).

El bootstrap [13], en términos de clasificación, consiste en replicar la totalidad de los experimentos de clasificación un número elevado de veces y estimar la solución usando el conjunto de dichos experimentos. Para estimar el porcentaje de error en un conjunto de datos de tamaño n se generan un gran número de muestras (seleccionando aleatoriamente) de igual tamaño (n) mediante muestreo con reemplazo a partir de dicho conjunto. Cada una de estas muestras se utiliza para construir una regla de clasificación que será usada para predicir la clase de aquellos datos originales que no fueron usados en el conjunto de entrenamiento. Esto da una estimación del porcentaje de error para cada muestra. La media de todas estas estimaciones es usada para calcular el porcentaje de error de la regla original. Debido a que este proceso está basado en la seleccion aleatoria existe la probabiliad de que haya algunos datos del conjunto original que no sean usados y otros que intervengan en más de un subconjunto. Por ello, es importante realizar un número de experimentos considerable para reducir la probabilidad de que existan datos del conjunto original que no hayan sido usados.

En este trabajo proponemos un método de selección de atributos que ordena descendentemente los atributos según su relevancia. Con él se pretendereducir el coste que implica el uso de un proceso de selección de atributos exahustivo sin perder la eficiencia del mismo. Para ello, utilizaremos una búsqueda aleatoria basada en las técnicas de bootstrapping (selección con reemplazo).

La búsqueda exhaustiva es inabordable para bases de datos que contengan un número considerable de atributos. Con lo que, para comparar el método que presentamos con el método exhaustivo, limitaremos dicha búsqueda atendiendo al número de atributos que intervienen en cada paso de ésta. Es decir, en vez de realizar todas las combinaciones posibles de atributos, se generarán aquellas combinaciones cuyo tamaño (número de atributos que intervienen en la combinación) sea equivalente a un valor preestablecido (K).

El documento se organiza de la siguiente forma. En la sección 2 se describe con detalle el método propuesta. En la sección 3 se muestra los resultados de los experimento y, por último, se muestran las conclusiones más interesantes son anotadas en la sección 4.

2 Descripción

El proceso de seleción de atributos que se propone puede ser dividido en cuatro fases: generación de subconjuntos de atributos (*generación*), evaluación de cada subconjunto (*evaluación*), actualización los pesos de cada atributo (*actualización*) y ordenación de atributos por su peso (*ordenación*). En la figura1 se representan estas fases sobre una base de datos ficticia de 5 atributos.

En la primera fase se generan los distintos subconjuntos de atributos que serán usados en la etapa posterior. Cada subconjunto contendrá un número máximo de atributos que serán escogidos aleatoriamente entre el conjunto de datos original, con la particularidad

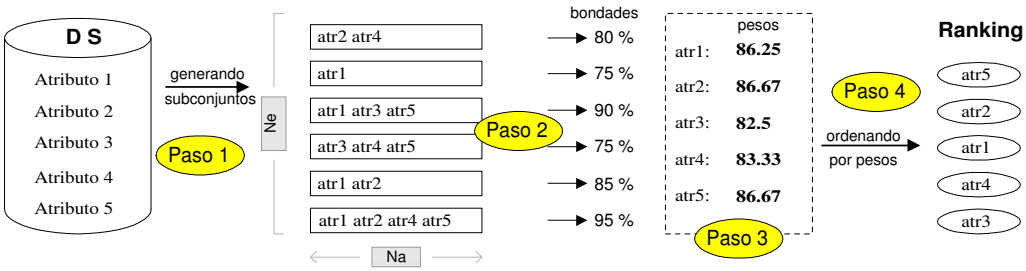

Figura 1: Proceso de Selección de Atributos propuesto.

de que en un mismo subconjunto no existan dos atributos iguales. Cada subconjunto será generado independientemente del anterior pudiendo existir dos iguales. Tanto el número de subconjunto generados (Ne) como el número máximo de atributos contenidos en éstos (Na) serán establecido por el usuario. En la figura1 el valor elegido es de 6 subconjuntos y 4 atributos como máximo, respectivamente. Los tres primeros subconjuntos generados han sido uno con dos atributos (atributo2 y atributo4), un segundo con uno (atributo1), un tercero con 3 (atributo1, atributo3 y atributo5).

La fase de evaluación consiste en clasificar el conjunto de datos original con cada uno de los subconjuntos de atributos generados en la etapa anterior. El resultado es asignar a cada subconjunto una bondad, que será el porcentaje de acierto de cada una de las clasificaciones. Como consecuencia de este paso, se le ha asignado al primer subconjunto de la figura 1 una bondad de 80. La mayor bondad generada ha sido de 95 para el último subconjunto, mientras que la menor, con valor 75, es compartida tanto para el segundo como para el cuarto.

En la fase de actualización se asignan los pesos de los atributos del conjunto de datos original. El peso de un atributo a (P_a) es la media de las bondades de los subconjunto que contengan a dicho atributo. Por ejemplo, el atributo1 de la figura1 forma parte del segundo, tercero, quinto y sexto subconjunto, con lo que su peso será: $P_{a_1} = \frac{b_2+b_3+b_5+b_6}{4} = 86.25$, donde b_i denota la bondad del subconjunto i.

El último paso consiste en organizar los atributos desdendentemente según su peso. Esta fase genera una lista que será el ranking devuelto por el proceso. En el ejemplo que nos ocupa, el ranking generado ha sido: atributo5, atributo2, atributo1, atributo4 y atributo3 como menos relevante.

La única y gran diferencia que presenta el método propuesto, descrito con los cuatro pasos anteriores, con respecto al método exhaustiva reside en la primera etapa. Por ejemplo, un metodo exhaustivo con $K = 1$ tendría que generar un subuconjunto por cada atributos del conjunto orignal. Sin embargo, si escogemos un valor de $K = 2$, se debería generar un subconjunto por cada par de atributos posibles. Por tanto, este primer paso para un método exhaustivo depende el valor de K, mientras que en el aleatorio depende del número de experimentos y del número máximo de atributos.

El pseudocódigo del método propuesto se muestra en la figura 2. El algoritmo se divide en dos funciones. La primera, GeneraRanking, es la función principal y tiene como parámetros de entrada: el método de clasificación que se utilizará para evaluar cada subconjunto de atributos escogido U; el conjunto de atributos a tratar X; el número de experimentos que se realizarán Ne y el número máximo de atributos que intervendrán en cada experimento Na. Esta función ofrece un único parámetro de salida (L), el cual corresponde con el ranking de atributos obtenido al aplicar el método de selección de atributos. Con tal fin, la función GeneraRanking ordena descendentemente los atributos según su peso. Para calcular este

```
Function GeneraRanking
   Inputs:
      U: Criterio Evaluación;
      X: Atributos;
      Ne: nº Experimentos;
      Na: nº Atributos
   Output:
      L: Lista Atributos (atributos con mayor
peso al principio)
 begin
   S := GeneraCombAtts(X, Ne, Na)
   foreach subconjunto de atributos S_i ∈ S
      C := Evalua(S_i, U)
      AcutalizaAtributos(S_i, C)
   end foreach
   L = Ordena(X)
end GeneraRanking

Function GeneraCombAtts
   Inputs:
      X: Atributos;
      Ne: Número Experimentos;
      Na: Número Máximo Atributos
   Output:
      L: Lista de subconjuntos de atributos
begin
   for i = 1 to Ne
      n := GeneraNumeroAleatorio(1, Na)
      S_i := EscogeAtributos(X, n)
      L := L + S_i
   end for
end GeneraCombAtts
```

Figura 2: Algoritmo para Generar Ranking.

valor, por cada atribuo a_i, se realiza la media de los porcentajes de acierto resultantes de aplicar el método de clasificacón U sobre aquellos subconjuntos que contengan al atributo a_i. Para obtener estos subconjuntos de atributos se hace uso de la función GeneraCombAtts (paso 1). En ésta, cada subconjunto es escogido aleatoriamente con un tamaño igualmente aleatorio (mayor que 1 y menor o igual que Na) de forma que no exista ningún atributo duplicado en un mismo subconjunto.

Según el proceso descrito anteriormente, podríamos enmarcar el método propuesto dentro de las técnicas de ranking basadas en wrappers.

3 Experimentos

En esta sección se mostrarán los resultados de los experimentos realizados. Con ellos se trata de comparar el método propuesto (búsqueda aleatoria) con el método exhaustivo. El conjunto de datos escogido para llevar a cabo la

```
Function EvaluaRanking
   Inputs:
      La: Lista Atributos(ranking a evaluar);
      U: Criterio Evaluación;
      Nae: nº Atributos a Evaluar
   Output:
      Le: Lista Evaluacion
 begin
   ListaAux := {}
   Le := {}
   for i = 1 hasta Nae
      at := atributo i-ésimo de L
      ListaAux := ListaAux + {at}
      C := Clasifica(ListaAtributos, U)
      Le(i) := C
   end for
end EvaluaRanking
```

Figura 3: Algoritmo para Evaluar Ranking.

comparativa se compone de cuatro bases de datos biológicas de expresión genómica cuyas características se muestran en la tabla 1. En éstas los atributos representan genes, y los ejemplos condiciones, de forma que en cada par (gen_i, $condicion_j$) se almacena el nivel de expresión del gen i-ésimo bajo la condición j-ésima. La particularidad que presentan es el elevado número de atributos frente al número reducido de ejemplos.

El proceso seguido para comparar dichos métodos consiste en evaluar el ranking obtenido en cada uno de ellos. Esta evaluación está basada en el cálculo del área existente bajo la curva de aprendizaje(AUC).

En terminos generales, cada punto (x, y) de la curva se calcula atendiendo al número de atributos que intervienen al clasificar el conjunto de datos original, donde 'x' representa al número de atributos e 'y' a la evaluación de la clasificación. El número de atributos seleccionados en el cálculo de cada punto es incrementado en uno hasta llegar al un número de atributos dado.

El pseudocódigo para evaluar el ranking se muestra en la figura 3. El algoritmo está compuesto por una sóla función denominada EvaluaRanking. Ésta posee tres argumentos de entrada: el ranking de atributos que se desea evaluar (La), el método de clasificación que se usará (U) y el número total de atributos que se utilizarán (Na). Como único argumento de salida presenta una lista (Le), la cual contiene

Tabla 1: Conjunto de datos usado.

			Características		
	Ref	**Acrónimo**	nºEj	nºAtr	clases
Dataset_C	[14]	dsc	60	7129	2
Colon	[15]	col	62	2000	2
Leucemia	[16]	leuk	38	7129	2
Linfoma	[17]	linf	96	4026	9

en cada una de sus posiciones los puntos que delimitan la curva. Éstos se calculan a partir del porcentaje de acierto obtenido en cada una de las clasificaciones (utilizando el criterio U), usando un número de atributos que aumenta en uno en cada clasificación. Comienza con el valor uno y finaliza con un número de atributos dado (Nae). Los atributos son escogidos según el orden indicado en el ranking que se quiere evaluar (La).

Una vez calculados dichos valores, se representa la curva de forma tal que en el eje de abscisa se indica el número de atributos escogidos y en el eje de ordenadas la evaluación de la clasificación obtenida para ese número de atributos.

Como se ha mencionado anteriormente, la curva de aprendizaje será usada para calcular el área existente bajo ella. Ésta estará normalizada, es decir, los valores existentes en el eje de abscisa se normalizan usando normalización lineal.

De esta forma un ranking será mejor que otro cuando el AUC sea mayor que el del otro.

En la tabla 2 se muestra el AUC (calculada según este proceso) tras evaluar los cien primeros atributos de los rankings obtenidos al aplicar el método de selección de atributos propuesto (método basado en bootstrapping) y el exhaustivo (con $K = 1$ y $K = 2$) sobre las base de datos mencionadas en la tabla 1.

La configuración elegida, tanto para los procesos de generación como para los de evaluación de los rankings, ha sido 3-NN ([18]) como método de clasificación y validación cruzada, usando cinco folds, como criterio de evaluación. El número máximo de atributos que intervienen en cada paso del método basado en bootstrapping ha sido limitado al $0,4\%$ [1] del número de atributos que posee la base de datos en cuestión. Este último método se ejecutará tres veces, todas ellas con la configuración mencionada, pero con un número de experimentos distintos. Este número será: equivalente al número de atributos que tenga la base de datos (M) para la primera ejecución($1 \times M$); el doble de los atributos que tenga para la segunda ($2 \times M$); y el triple para la tercera ejecución ($3 \times M$).

Nótese que el número de pasos realizados en los métodos con búsqueda exhaustiva depende tanto de la cantidad de atributos que posea la base de datos en cuestión como del valor de K, mientras que en el propuesto tan sólo depende del número de experimentos que se desea realizar. En los experimentos que nos ocupan, dicho valor obedece al número de atributos de las base de datos en cuestión. Este razonamiento nos lleva a afirmar que nuestro método tiene un coste (número de pasos) aditivo, en lugar de multiplicativo como tiene el exhaustivo.

En la tabla 2, además de mostrar el resultado de la evaluación del ranking generado por los distintos métodos, se presenta el número de pasos realizados por cada uno de ellos. En esta tabla igualmente, se resalta en negrita el mejor AUC obtenida para cada base de datos. Por tanto, para la base de datos dsc (tumor del sistema nervioso), por ejemplo, el método

[1] Se escoge un valor pequeño de Na para que los subconjuntos generados en cada paso no posean un gran número de atributos. De esta forma reducimos la posiblidad de escoger atributos irrelevantes con otros que si lo son y, por tanto, no asignar a un atributo un peso mayor que el que le corresponde por el hecho de haber sido seleccionado junto a otros atributos que son más relevants que él. Es decir, con este valor conseguimos atenuar los efectos de la aletoriedad

Tabla 2: Resultados experimentales.

	Búsqueda exhaustiva				Búsqueda aleatoria					
	K=1		*K=2*		*1 * M*		*2 * M*		*3 * M*	
BD	nºPasos	AUC	nºPasos	AUC	nºPasos	AUC	nºPasos	AUC	nºPasos	AUC
dsc	7.129	67,5	25.407.756	83,39	7.129	81,83	14.258	83,73	21.387	**84,1**
col	2.000	82,77	1.999.000	**84,23**	2.000	83,05	4.000	83,87	6.000	84,06
leuk	7.129	95,02	25.407.756	95,53	7.129	96,22	14.258	96,48	21.387	**96,69**
linf	4.026	78,84	8.102.325	78,82	4.026	84,98	8.052	86,43	12.078	**87,47**

exhaustivo $K = 1$ ha necesitado 7.129 pasos para obtener un resultado de $67,5$; mientras que la siguiente ejecución posible para este método ($K = 2$) ha necesitado 25.407.756 pasos para superarlo con un valor igual a $83,39$. Si atendemos a los resultados obtenidos por el método propuesto, observamos que realizando el mismo número de pasos que $K = 1$ conseguimos un valor $(81,83)$ extremadamente superior al obtenido por éste y muy cercano al logrado por $K = 2$. Para alcanzar un valor igual o superior a este último han sido necesario 14.258 pasos, logrando un resultado de $83,73$. Es decir, tan sólo han sido necesario el $0,056\%$ de pasos para superarlo. El mejor resultado conseguido para la base de datos dsc ha sido $84,1$. El cual fue logrado por el método basado en búsqueda aleatoria ($3 \times M$), realizando 21.387 pasos para conseguirlo.

Comparando los resultados obtenidos por el método propuesto, apreciamos que son mejores cuantos más pasos realice éste.

Atendiendo a los mejores resultados obtenidos para cada base de datos, podemos observar que para todas ellas, menos para col, lo ha conseguido el método $3 \times M$. El mejor resultado conseguido con esta última base de datos ha sido el obtenido por el método $K = 2$. Para explicar el motivo de este suceso debemos estudiar las dependencias que tiene nuestro método con sus dos únicos valores de entrada (número de pasos a realizar y número máximo de atributos en cada paso). De todas formas, nótese que $K = 2$, para conseguir el valor de $84,23$, ha precisado de 1.999.000 pasos, mientras que $3 \times M$ ha conseguido un valor $(84,06)$ muy cercano con el $0,3\%$ de pasos.

Con el fin de estudiar las dependencias de nuestro método con sus valores de entrada se han realizado diferentes experimentos sobre las base de datos mencionadas en la tabla 1. Los cuales han consistido en evaluar los rankings obtenidos tras aplicar el método basado en bootstrapping variando los datos de entrada de éste. La configuración utilizada, tanto para la evaluación como para la generación de ranking, ha sido la misma que la utilizada en los experimentos anteriores, sólo que en este caso, el número máximo de atributos en cada paso (Na) ha sido variado, entre 1 y 5, en incrementos de $0,1$. Al haber aplicado el estudio tres veces por cada base de datos, cada una de ellas variando el número de pasos realizados en $1 \times M$, $2 \times M$ y $3 \times M$, respectivamente, el número de experimentos efectuados asciende a 6.085.200 ((1+2+3)* nº att en cada BD * 50 "nº diferentes Na utilizados"), como hemos realizado validación cruzada usando 5 folds en cada experimento, el total de modelos generados ha sido 30.426.00 (5 folds * nº Experimentos).

En resumen, se han obtenido 600 rankings diferentes que han sido evaluados mediante el algoritmo ilustrado en la figura 3. Los resultados de las evaluaciones vienen representados en la figura 4, la cual está dividida en cuatro gráficas, cada una ellas correspondiente a los resultados sobre una base de datos. Estos resultados son mostrados mediante las tendencias de las evaluaciones de los rankings obtenidos ante las variaciones de Na, de forma tal que en el eje de abscisas se indica el número máximo de atributos utilizados en la ejecuión del método y en el eje de ordenadas el AUC obtenido tras evaluar el ranking generado en dicha ejecución. El estudio de estas tendencias ha sido dividido según el número de pasos realizados por el método propuesto, de manera que en cada una de las gráficas existen tres curvas correspondientes a las tendencias de la ejecución del método con $1 \times M$, $2 \times M$ y $3 \times M$

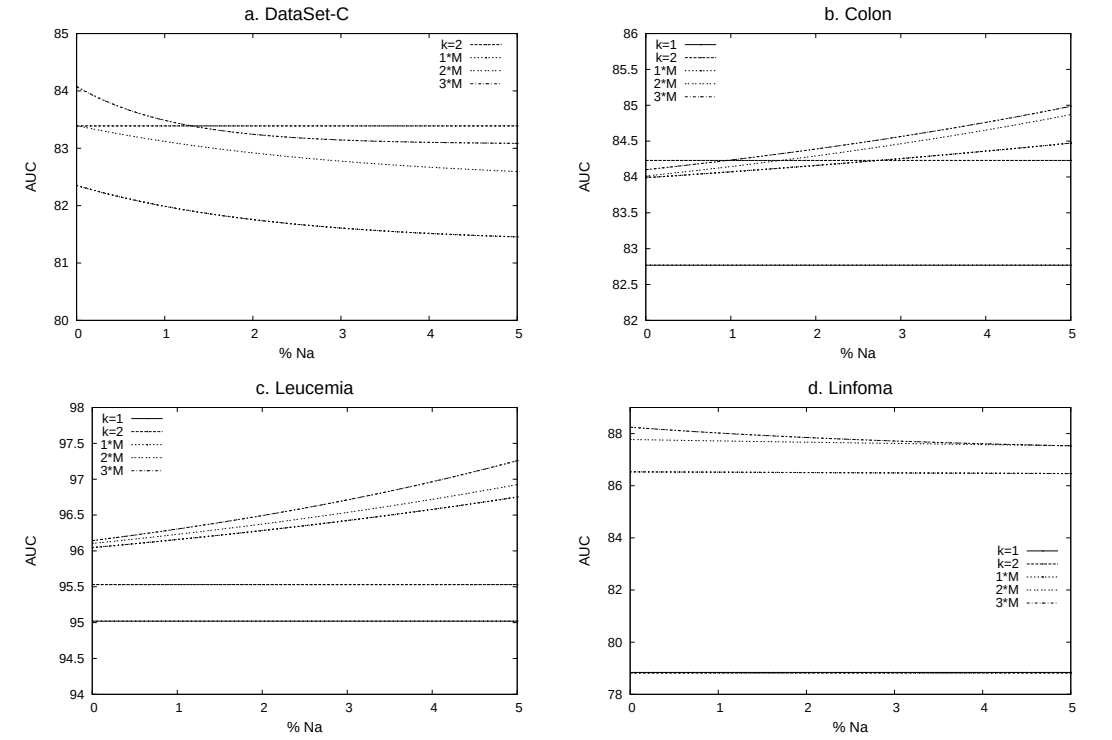

Figura 4: Tendencias.

experimentos respectivamente.

A parte de dichas curvas, para poder comparar dicho método con el exhaustivo, en cada gráfica se representa las tendencias del AUC resultado de evaluar los rankings obtenidos tras aplicar el método exhaustivo con $K = 1$ y $K = 2$. La configuración usada en éstos, tanto para la generación como para la evaluación, ha sido equivalente a la utilizada anteriormente. Nótese que los rankings generados por el método exhaustivo no dependen de Na y, por consiguiente, sus tendencias serán representadas por recta con pendiente 0. Cada una de estas rectas, por tanto, serán paralelas al eje de ordenadas y cortarán al eje de abscisas en el punto que tiene el mismo valor que el AUC mostrado en la tabla 2 en las columnas "K=1" y "K=2" para cada base de datos.

Hemos de señalar que la tendencia correspondiente a $K = 1$ para DataSet-C no ha sido representada debido a que en caso de hacerlo no se apreciaría con la misma claridad las tendencias del método propuesto, al tener que agrandar el rango de valores del eje OY por encontrarse dicha recta en $y = 67,5$. También tenemos que remarcar el hecho de que las tendencias de $K = 1$ y $K = 2$ para Linfoma se solapan una con la otra ($78,84 \simeq 78,82$).

A partir de dichas gráficas, observamos que dependiendo de la base de datos escogida las tendencias del método propuesto son crecientes o decreciente. Este suceso puede ser debido a diferentes circunstancias. Una de ellas es el grado de correlación que exista entre los diferentes atributos de la base de datos, de forma que con una gran correlación la pendiente será decreciente, y creciente en caso contrario. A partir de este razonamiento, el número Na debe ser inversamente proporcional a dicho grado de correlación.

Atendiendo al otro valor de entrada de nuestro método (Ne), podemos decir que cuanto mayor sea éste mejor resultado obtendremos. Esto es así puesto que los valores de las tendencias de dicho método son superiores cuanto más pasos realice éste.

Teniendo en cuenta esta figura, también es posible comparar la eficiencia del método basado en bootstrapping con el exhaustivo. Sabiendo que un método es más eficiente que otro si su tendencia está por encima, podemos observar que tanto para la base de datos Leucemia como para Linfoma el método basado en bootstrapping es superior al exhaustivo, mientras que para las otras dos restantes depende del número máximo de atributos escogido (Na). En caso de basarnos en la base de datos DataSet-C para comparar estos métodos, podemos decir que el método propuesto será más eficiente que $K = 2$ tan sólo en el caso de elegir un número de experimentos $3 \times M$ y establecer como Na un número menor que $1,3\%$ del número total de atributos de dicha base de datos. Si por el contrario nos basamos en Colon para realizar dicha comparativa, se observa que nuestro método será superior al exhaustivo en caso de escoger para Na un valor superior al $2,8\%$ del número de atributos de ésta, independientemente del número de pasos a realizar. En caso de escoger $Ne = 3 \times M$ en esta última comparativa, el valor de Na podría tomar valores superiores a $0,9\%$.

Para demostrar que a partir de la posición de las tendencias se pueden comparar las eficiencias de los distintos métodos se presenta la figura 5. En ella se muestran las curvas delimitadoras del AUC, resultado de aplicar el algoritmo de evaluación mostrado en la figura3 sobre los rankings generados por los distintos métodos (aleatorio con $1 \times M$, $2 \times M$ y $3 \times M$; y exhaustivo con $K = 1$ y $K = 2$) sobre la base de datos DataSet-C. Es decir, en el eje de ordenadas se encuentra el número de atributos escogidos (según el orden indicado en el ranking a evaluar) y en el de abscisas el porcentaje de acierto de la clasificación.

En la figura 5 se muestran dos gráficas cuya configuración de dichos métodos tan sólo difieren en el Na escogido, el resto de ésta es la misma que la utilizada en los experimentos anteriores. Los valores elegidos para Na han sido tales que la curva representante del método $3 \times M$ fuera en un caso superior a la representante de $K = 2$ y en otro inferior, en concreto, $0,4$ y $4,3$ respectivamente. De esta forma se puede apreciar que las clasificaciones del método propuesto al evaluar los 100 primeros atributos en comparación a las clasi-

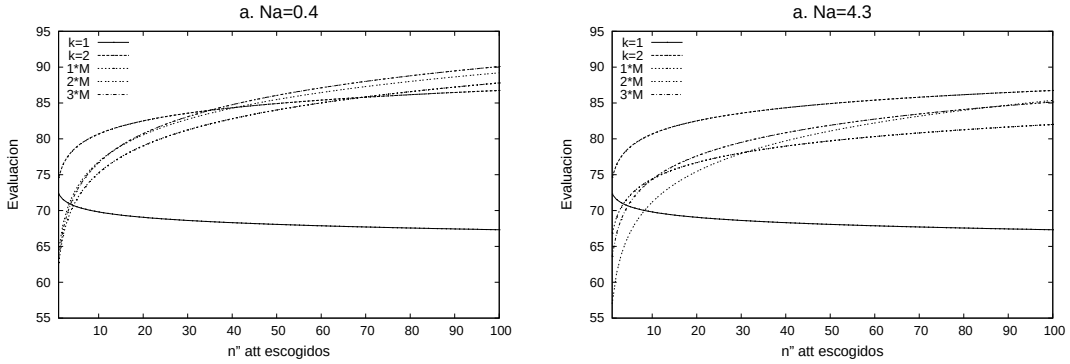

Figura 5: DataSet-C. Curvas Delimitadoras.

ficaciones obtenidas por el método $K = 2$ son acordes con la posición de las tendencias ya que son superiores en el caso en que $Na = 0,4$ e inferiores en el otro.

Además de lo mencionando, se sigue apreciando que cuanto mayor sea el número de experimentos que realice el método propuesto mejor será el resultado obtenido.

En resumen, observando las comparaciones realizadas entre el método basado en bootstrapping con el exhaustivo, podemos afirmar que el método propuesto genera rankings de atributos cuyas evaluación equipara e incluso mejora a los generados por el método exhaustivo y, además, son generados en un número muy inferior de pasos.

4 Conclusiones

En este documento presentamos un nuevo algoritmo de selección de atributos cuyo objetivo es reducir el coste computacional de un método de selección exhaustivo sin perder su eficacia. La búsqueda llevada a cabo por el nuevo método se basa en las técnicas de bootstrapping, las cuales realizan una selección aleatoria con reemplazo.

Tras los experimentos empíricos, concluimos que nuestra propuesta produce resultados similares al método exhaustivo en un número muy reducido de pasos. Esta reducción oscila entorno a la ratio 1:100.000, es decir, por cada 100.000 iteraciones nuestro método utiliza sólo una.

References

[1] M. Dash and H. Liu *Feature Selection for Classification.* Intelligent Data Analysis, Elsevier, Vol. 1, no. 3, pp 131 - 156, 1997

[2] H. Liu and Lei Yu *Toward Integrating Feature Selection Algorithms for Classification and Clustering.* IEEE Trans. on Knowledge and Data Engineering, vol. 17, no. 4, pp 491-502, 2005

[3] H. Liu et al. *Feature Selection with Selective Sampling.* Proc. 19th Int´l Conf. Machine Learning, pp. 395-402,2002

[4] C. Cardie *Using Decision Trees to Improve Case-Based Learning.* Proc. 10th Intl´l Conf. Machine Learning, P. Utgoff, ed., pp. 25-32, 1993.

[5] A.N. Mucciardi and E.E. Gose *A comparison of Seven Techinques for Choosing Subsets of Pattern Recognition.* IEEE Trans. Computer, vol. 20, pp. 1023-1031,1971.

[6] R. Ruiz, J. Riquelme, J. Aguilar-Ruiz *Projection-based measure for efficient feature selection.* Journal of Intelligent and Fuzzy System, vol. 12,pp 175-183, 2003

[7] R. Kohavi and G.H. John *Wrappers for Feature Subset Selection.* Artificial Intelligence, vol. 97, no.1-2, pp. 273-324, 1997

[8] Jennifer G. Dy et al. *Feature Selection for Unsupervised Learning.* J. Mach. Learn. Res., vol. 5, pp. 845-889, 2004

[9] S. Das *Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection.* Proc. 18th Int´l Conf. Machine Learning, pp. 74-81, 2001

[10] J. Doak *An Evaluation of Feature Selection Methods and Their Apllication to Computer Security.* Technical report, Univ. of California at Davis, Dept. Computer Science,1992.

[11] H. Liu, H. Motoda *Feature Selection for Knowledge Discovery and Data Mining.* Boston: Kluwer Academic, 1998.

[12] G. Brassard et al. *Fundamentals of Algorithms.* New Jersey: Prentice Hall,1996.

[13] Efron, B. *Estimating the error rate of a prediction rule: improvements on cross-validation* J.Amer.Stat.Ass. vol 78, pp. 316-331, 1983

[14] Scott L. Pomeroy, et al *Prediction of Central Nervous System Embryonal Tumour Outcome based on Gene Expression.* NATURE, vol 415, pp. 436-442, 2002.

[15] U. Alon et al. *Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays.* PNAS, Vol 96, Issue 12, pp. 6745-6750, 1999.

[16] T.R. Golub, et al. *Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring.* SCIENCE, vol 286, pp. 531-537, 1999.

[17] Ash A. Alizadeh, et al *Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling.* NATURE, VOL 403, N^{o} 3, pp. 503-511, February 2000.

[18] Dasarathy, Belur V. *Nearest neighbor(NN) norms: NN pattern classification techniques.* IEEE Comp. Soc. Press. 1990.