Minería de Datos: Líneas de Investigación actuales en la Universidad de Sevilla

J.S. Aguilar-Ruiz, J.L. Álvarez, F. Ferrer-Troyano, R. Giráldez,
J. Mata, D. Mateos, J.C. Riquelme, R. Ruiz, A. Troncoso

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
riquelme@lsi.us.es

Abstract. En este trabajo se pretende describir resumidamente las líneas de investigación que actualmente están abiertas en Aprendizaje Automático y Minería de Datos en nuestro grupo. Asimismo se resumirán los proyectos de investigación y líneas de colaboración establecidas con otros grupos.

1 Introducción

En la actualidad el grupo de profesores del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Sevilla dedicado a investigaciones próximas a la Minería de Datos está compuesto por dos doctores, dos doctores con las tesis depositadas y cuatro doctorandos con las tesis en fases avanzadas y un recién titulado. Desde el año 1995 la principal línea de investigación ha estado basada en la aplicación de algoritmos evolutivos a distintos campos del aprendizaje supervisado: obtención de nuevas características para clasificación [1], clasificadores basados en descripciones cualitativas [2], obtención de reglas de decisión jerárquicas [3], etc.

Con la incorporación de nuevos miembros al grupo, en el año 1999 se abrieron nuevas líneas de trabajo abarcando distintas metodologías y fases dentro de un proceso de extracción de conocimiento en bases de datos. Así se investigaron técnicas para la reducción o editado del conjunto de instancias [4][5], técnicas de discretización [6], clustering supervisado [7], obtención de reglas de asociación en aprendizaje no supervisado [8], mejoras a la técnica k-NN [9] o distintas representaciones para las fronteras de decisión en clasificadores [10][11][12].

En este trabajo, nos vamos a centrar en las líneas de investigación actualmente en fase de trabajo. Abarcan también distintos aspectos de la minería de datos y aplicaciones a diferentes problemas reales. Por el orden en el que aparecen en el texto, se detallan metodologías para un procesado más eficiente de los datos durante un proceso evolutivo para la obtención de reglas, una técnica de selección de características, métodos de tratamiento de grandes cantidades de datos, visualización de reglas y aplicaciones a bases de datos provenientes de campos diversos: bioinformática, empresas eléctricas, proyectos de desarrollo de software, etc.

2 Aprendizaje evolutivo

Esta línea de trabajo está centrada en la generación de reglas de decisión en aprendizaje supervisado aplicando algoritmos evolutivos, y más concretamente en la mejora de dos aspectos fundamentales de dichos algoritmos: la eficiencia (acelerando el proceso evolutivo y reduciendo espacio de almacenamiento) y la eficacia (generando modelos de conocimientos cada vez más precisos y más sencillos de comprender). El resultado fundamental de nuestra investigación es la herramienta denominada HIDER (Hierarchical Decision Rules) [13], cuyas diferentes versiones han aportado mejoras sobre otros sistemas respecto a los dos aspectos antes citados.

2.1 Obtención de las reglas

HIDER aplica un Algoritmo Evolutivo de cubrimiento secuencial para la generación de reglas de decisión jerárquicas en aprendizaje supervisado, donde el espacio de búsqueda lo forman las reglas de decisión que son codificadas como individuos de la población genética. HIDER tienen el mismo objetivo que herramientas como C4.5 y sus variantes, sin embargo, es muy diferente tanto en las reglas que obtiene como en la metodología que sigue para ello. HIDER encuentra un conjunto jerárquico de reglas que facilita su comprensión por dos motivos: además de generar un número de reglas significativamente menor que C4.5, la jerarquía proporciona una información añadida sobre la inclusión espacial de las regiones que el C4.5 no aporta. Respecto a la metodología usada por cada método, la búsqueda de fronteras de decisión es la que marca la diferencia fundamental para obtener mayor precisión con menor número de reglas. Mientras C4.5 es un proceso recursivo que en cada paso establece una única frontera para un solo parámetro o atributo, HIDER encuentra simultáneamente dos fronteras para todos los atributos. Es decir, C4.5 establece en un momento dado una condición sobre un atributo porque en ese instante entiende que es la mejor, sin tener en consideración que en el proceso posterior de establecer condiciones sobre los demás atributos, esa primera opción pudiere no ser la mejor. HIDER trabaja sobre todos los atributos a la vez en cada proceso evolutivo, de modo que un intervalo determinado para un atributo puede ser modificado en una generación posterior en función de los intervalos establecidos para el resto de atributos.

2.2 Mejoras en eficiencia

Un gran número de métodos clásicos de aprendizaje automático utilizan algoritmos probabilísticos para buscar soluciones en el espacio, con un alto coste computacional, principalmente por la repetitiva evaluación de las soluciones candidatas. En concreto, HIDER emplea el 85% del tiempo en evaluar los individuos de las poblaciones genéticas que va generando a lo largo del proceso de aprendizaje. Esto nos impulsó a desarrollar un estructura de datos que acelerara el proceso de evaluación de individuos. Hasta ese momento, cada individuo se evaluaba recorriendo la base de datos lineal-

mente (esté ésta en disco o cargada en memoria) y contando los ejemplos que dicho individuo cubría. Dada la estructura y la semántica de las reglas de decisión, esta evaluación se puede llevar a cabo de un modo más eficiente. Para ello, los ejemplos de la base de datos se indexan usando la estructura EES (Efficient Evaluation Structure). Esta estructura distribuye la información de la base de datos en un vector de árboles binarios de búsqueda. Cada celda del vector contiene la información sobre un atributo de la base de datos, en concreto un árbol de búsqueda en el que cada nodo contiene uno de los valores que el atributo puede tomar así como una lista de los índices de los ejemplos que toman dicho valor para el atributo correspondiente. Así, durante la evaluación de una regla, se toman las condiciones que dicha regla establece para cada atributo y sólo se consultan aquellos ejemplos cuyos valores son cubiertos por tales condiciones, realizando búsquedas en los árboles, y no la totalidad de la base de datos.

Con el propósito de simplificar el espacio de búsqueda y subsanar los inconvenientes que otras codificaciones como la binaria o la real presentan, desarrollamos la denominada Codificación Natural [14]. En esta codificación, al contrario que otras codificaciones, cada atributo es codificado por un único gen, que es un número natural. Además, los operadores genéticos (cruce y mutación) se reducen a expresiones algebraicas simples cuyo coste de aplicación es O(1). Así, la codificación natural aumenta la eficiencia del algoritmo evolutivo, ya que disminuye el espacio de búsqueda respecto a la codificación real, cuyo espacio es infinito, convirtiéndolo en un conjunto de valores naturales finitos. Este aspecto, por otro lado, aumenta también la eficacia del método, ya que el espacio puede ser explorado más ampliamente y las reglas resultantes tienden a ser más precisas.

Otra ventaja de esta codificación se presenta al combinarla con la estructura EES, dando como resultado la EES^{cn}, donde los árboles de búsqueda han sido sustituidos por tablas Hash. La codificación natural permite realizar un acceso directo a las celdas de esas tablas mediante una simple operación aritmética sobre los genes del individuo. Esto reduce el coste logarítmico de la búsqueda en el árbol a un coste unidad.

3 Selección de características

La selección de características (feature selection) consiste básicamente en una disminución del espacio de parámetros o atributos que intervienen en la base de datos. Existen muy diversas formas de catalogar a los algoritmos de selección de atributos, en función de la característica que queramos acentuar. Por el punto de partida: forward o backward, según empiecen desde cero o con todos los atributos respectivamente, y a veces mezcla de ambos. Por el tipo de búsqueda: exhaustiva o heurística. Por la estrategia de evaluación: wrappers o filtros. Por el criterio de parada: donde se decide el momento en que debe terminar el algoritmo, puede ser al alcanzar un número determinado de atributos, cuando se realice cierto número de iteraciones, se detecte que no hay mejora, o se avance hasta una exactitud prefijada. Por la estrategia de evaluación: wrappers, que utilizan el propio algoritmo de clasificación para evaluar la utilidad de

los parámetros, o filtros que evalúan los atributos de acuerdo con heurísticas basadas en características generales de los datos e independientes del método de clasificación a aplicar. Además, se pueden utilizar procedimientos de aprendizaje para seleccionar atributos, métodos basado en la correlación, etc.

3.1 Algoritmo SOAP

Nuestra línea de trabajo intenta buscar un método de selección de características que sea lo más rápido posible dentro de unos criterios de bondad aceptables. La razón para ello es tener la posibilidad de utilizar un algoritmo evolutivo o incluso búsquedas exhaustivas para un refinamiento posterior, tanto de los atributos originales como de combinaciones aritméticas o lógicas de ellos, que permitan la creación de nuevas características en función de las originales.

Para explicar el método se ha realizado una proyección de la conocida base de datos Iris en un gráficos bidimensional. Así, en la figura 1 se puede observar como para el parámetro Sepalwidth (en el eje de ordenadas) no se aprecian intervalos nítidos y sin embargo, para el atributo Petalwidth se pueden advertir algunos intervalos donde la clase es única: [0,0.6] para Setosa, [1.0,1.3] para Versicolor y [1.8,2.5] para Virginica. Esto es debido a que al proyectar las etiquetas de las clases sobre cada atributo el número de cambios de etiqueta es menor en el segundo caso. Por ejemplo, se puede comprobar que para Petalwidth el primer cambio de etiqueta se produce para el valor 1 (de setosa a Versicolor), el segundo en 1.3 (de Versicolor a Virginica), y desde 1.8 hasta el final sólo nos encontramos individuos de la clase Virginica.

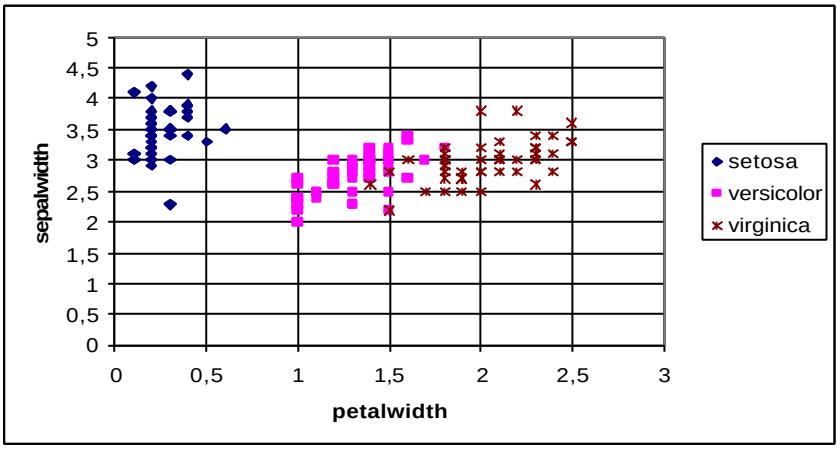

Fig. 1. Proyección de la Base de Datos IRIS

En este sencillo principio se basa SOAP [15] (Selection Of Attributes by Projection): contar los cambios de etiquetas de los ejemplos, producidos al recorrer las proyecciones de cada uno de ellos en cada dimensión. Si posteriormente se ordenan ascendentemente los atributos según el número de cambios de etiquetas tendremos una lista

ordenada que define el orden de selección de mayor a menor importancia. SOAP supone eliminada la redundancia básica entre atributos. Finalmente para escoger el número de parámetros más conveniente, definimos un factor de reducción tal que tomaremos el subconjunto de atributos formado por los primeros de la lista antes mencionada.

3.2 Pruebas realizadas

Para estudiar si la reducción de parámetros propuesta por SOAP afecta a la bondad de una clasificación posterior, hemos aplicado el C4.5 como técnica de clasificación antes y después de la reducción. Normalmente la reducción de parámetros afectará negativamente las tasas de acierto pero se pretende que la perdida de información no sea excesiva. Por ello, se comparan los resultados antes y después de la reducción con SOAP, ReliefF y CFS, validando los resultados con las BD originales y comparando los métodos de reducción entre sí. Los resultados obtenidos al procesar 16 BD, muestran que en 7 la reducción efectuada por SOAP afecta negativamente la exactitud con respecto a la BD original, y en 9 no existe pérdida de información. Estos resultados son similares a los obtenidos por ReliefF y un poco peores que los proporcionados por CFS. Sin embargo, aunque la reducción de la exactitud es estadísticamente significativa, hay que señalar que tampoco es considerable. SOAP es un algoritmo que destaca especialmente por necesitar un tiempo de computación muy reducido.
SOAP es un algoritmo de selección de atributos muy eficiente y simple, utilizado en la fase de preprocesado para reducir de manera considerable el número de atributos. Todo ello sin la necesidad de realizar cálculos estadísticos que podrían ser muy costosos en tiempo (correlación, ganancia de información, etc.), siendo el nuestro equivalente a la ordenación más rápida O(M?N?log N). Al mismo tiempo se mantienen las tasas de error, y se obtienen unos modelos de clasificación mucho más simples.

4 Data Streams

4.1 Introducción

El procesamiento de enormes cantidades de información al que se enfrentan las grandes organizaciones ha transformando los estándares tradicionales de la Minería de Datos haciendo que en la extracción de conocimiento no sólo participen las bases de datos operacionales de la propia organización. Debido a que estos datos se encuentran usualmente sometidos a un tráfico permanente, son altamente susceptibles al ruido, la perdida y la inconsistencia en sus valores. Por esta razón una predicción muy precisa de la tendencia o las necesidades del mercado puede no ser tan importante como un modelo útil que sirva de apoyo a las decisiones del gestor.

Sin embargo las limitaciones en tiempo y memoria convierten en paradójica la disponibilidad de cantidades de datos tan enormes: cuando no es posible cargar en me-

moria todos los registros o ejemplos de una tabla o consulta sobre la que obtener patrones de comportamiento, la mayoría de los sistemas de aprendizaje tradicionales deben ser aplicados a subconjuntos más pequeños generados mediante técnicas de muestreo. Una propuesta estudiada por varios autores consiste en combinar los submodelos extraídos de dichos subconjuntos con objeto de obtener el modelo global. A través de este mecanismo pueden cometerse fácilmente errores de sobreajuste por los cuales patrones válidos para pequeños subconjuntos pueden no serlo para todo el conjunto.

Tal situación ha provocado la necesidad de diseñar algoritmos de aprendizaje incrementales que actualizan el modelo mientras procesan los registros secuencialmente a una gran velocidad, algoritmos a su vez escalables, capaces de minar millones de ejemplos sin que el orden de llegada de los mismos afecte de forma significativa el modelo generado, y algoritmos que permitan obtener un modelo de salida en todo momento, aunque no se hayan leído todos los registros. Hasta la fecha, muchos algoritmos de aprendizaje escalables e incrementales se han basado en árboles de decisión. Esta aproximación puede causar dos problemas. El primero es que ante la dimensionalidad y numerosidad de los datos de entrada el árbol inicial puede tener un tamaño que lo haga incomprensible al analista, gestor o experto del conocimiento. Para reducir dicho tamaño, la técnica de poda empleada puede requerir un tiempo prohibitivo en este contexto. En el minado de flujos de datos transaccionales, uno de los principales objetivos debe ser hacer posible que cada nuevo ejemplo actualice el modelo antes de que llegue el siguiente. Con datos en los que la distribución en el tiempo no es estática, aparece el segundo problema: subárboles del árbol global pasan a ser obsoletos, con lo que actualizar el árbol global puede incrementar de nuevo el coste computacional.

Nuestra línea de trabajo consiste un algoritmo de aprendizaje que recibe el nombre AAFT (Algoritmo de Aprendizaje para Flujos de Datos Transaccionales) y obtiene reglas de decisión leyendo una sola vez registros procedentes de bases de datos transaccionales. Las reglas van siendo eliminadas cuando se convierten en obsoletas mientras nuevos ejemplos quedan aún por procesar. Tal reducción del modelo no sólo no daña el coste computacional sino que acelera el proceso de actualización. Con la utilidad del modelo como principal objetivo, proponemos una técnica que se separa de las existentes en la literatura ya que no necesita leer bloques de varios registros para continuar ni tampoco intenta modelar todo el espacio de búsqueda, sino únicamente las regiones de mayor influencia.

4.2 Descripción de AAFT

El modelo generado por AAFT es un conjunto de reglas cada una de ellas conjuntos a su vez de la forma $R=\{I,C,V,E,S,L\}$. $I=\{I_1,\ldots,I_m\}$ es un conjunto de m pares ordenados que representan intervalos cerrados $Ii=[I_{il}, I_{iu}]$ para cada atributo o dimensión relevante. Los límites inferior I_{il} y superior I_{iu} de cada intervalo son vértices de un hipercubo de aristas paralelas a los ejes con C como centro de masa. V es el volumen y define la región de influencia de la regla R. Decimos que una regla R cubre a un ejem-

plo cuando dicho punto pertenece al hipercubo o zona de influencia de *R*. *C* es un vector sintético (un prototipo o caso no real) en R^m que representa el centro de masa de los ejemplos procesados hasta el momento y que han sido cubiertos por una regla. $E=\{e_n,e_f\}$ son dos ejemplos reales cubiertos por *R* hasta el momento: el ejemplo más cercano e_n a *C* y el ejemplo más distante e_f a *C*. *S* es un peso llamado soporte de valor entero asociado a cada regla igual al número de ejemplos cubiertos por la misma hasta el momento. Finalmente *L* es la etiqueta asociada a *R*.

Cada vez que un nuevo ejemplo es leído de la fuente de datos de entrada, se actualiza el conjunto de reglas obtenido hasta ese momento de forma que cada cierto número de ejemplos leídos realiza una poda del conjunto de reglas guardando únicamente las de mayor soporte. También se eliminan atributos o dimensiones no relevantes. Hasta que no es leído el último ejemplo todas las reglas son disjuntas para etiquetas diferentes, es decir, sólo reglas con la misma etiqueta asociada pueden compartir regiones comunes.

Cuando un nuevo ejemplo es leído, se buscan todas las reglas que lo cubren. Si hay más de una regla que lo cubra, todas ellas tendrán la misma etiqueta asociada. En tal situación se distinguen dos casos:

- El nuevo ejemplo y las reglas que lo cubren tienen la misma etiqueta. En tal caso todas esas reglas son actualizadas con dos operaciones muy sencillas: el cálculo de su nuevo centro de masa y el incremento en una unidad de su soporte.
- El nuevo ejemplo tiene asociada una etiqueta diferente a la de las reglas que lo cubren. Entonces la regla cuyo centro de masa es el más cercano al nuevo ejemplo es dividida en tres nuevas reglas de volumen 0 de forma que para cada intervalo el límite inferior es igual al límite superior. La tercera regla es generada para cubrir el nuevo ejemplo (tendrá asociada por tanto la etiqueta de éste) y se le asigna el valor 1 como soporte. Las dos primeras son generadas para cubrir a e_n y e_f respectivamente, tienen la misma etiqueta que la regla original pero su soporte es inversamente proporcional a la distancia de cada uno de ellos respecto al centro de masa de la misma. Cuánto más cerca al centro de masa mayor será el soporte asignado.

Aunque el nuevo ejemplo puede estar cubierto por varias reglas con una etiqueta diferente, únicamente la regla con el centro de masa más cercano es eliminada. Hemos considerado tal criterio bajo la hipótesis de que si el nuevo ejemplo forma parte realmente de un patrón, entonces tarde o temprano se leerán nuevos ejemplos cercanos a él con la misma etiqueta que dividirán reglas erróneas. Nuestro argumento se basa en la alta susceptibilidad al ruido de los flujos de datos transaccionales. Si el nuevo ejemplo leído no forma parte de un patrón, dividir todas las reglas que lo cubren y que tienen etiqueta diferente a dicho ejemplo puede implicar un coste computacional innecesario, y además, provocar errores de sobreajuste.

Cuando no existe ninguna regla que cubre al nuevo ejemplo, el algoritmo busca la regla ampliable más cercana con igual etiqueta. Un hipercubo puede ampliarse para cubrir un nuevo ejemplo si no intersecta con ningún otro hipercubo con diferente etiqueta. Si dicha regla es encontrada, entonces es actualizada calculando su nuevo centro de masa, incrementando en uno el soporte asociado y aumentando su volumen para cubrir a dicho ejemplo. Cada cierto número de ejemplos se realiza una poda en el

modelo. El objetivo es eliminar outliers, ruido y atributos irrelevantes. Tras leer el último ejemplo de la fuente de datos se realiza una poda final.

5 Visualización

5.1 Introducción

Las técnicas de visualización de grandes almacenes de datos, entendidos como conjuntos potencialmente infinitos de elementos de múltiples variables (las cuales pueden ser de diferente naturaleza entre sí), se han constituido como una pieza clave a lo largo de todo el proceso KDD. La Minería de Datos Visual se sitúa en las dos últimas etapas del proceso KDD: Aprendizaje y Evaluación. Pueden diferenciarse tres tipos de aplicación para la Minería de Datos Visual dentro del proceso KDD:

- **Visualización del resultado final**. Un algoritmo de aprendizaje independiente obtiene patrones de comportamiento y éstos son representados posteriormente para ser interpretados. En base a dicha representación el usuario puede volver a la fase de aprendizaje y reajustar los parámetros del algoritmo de aprendizaje.
- **Visualización de un resultado intermedio**. Mediante un algoritmo independiente se lleva a cabo un análisis topológico del dominio y éste es visualizado. En base a dicha representación el usuario deduce posibles patrones.
- **Visualización de los datos**. Los datos son representados sin aplicar previamente un algoritmo de análisis o de aprendizaje. Reajustando parámetros e interactuando con la representación obtenida, los patrones son obtenidos directamente por el usuario mediante navegación y exploración de los datos.

La cardinalidad, dimensionalidad y heterogeneidad de los volúmenes de información que se manejan en las grandes organizaciones sigue siendo un reto en todas las líneas de la Minería de Datos. Con una tecnología disponible para generar, procesar y almacenar gigabytes diariamente, el hueco entre la cantidad de datos que se desea representar y la cantidad que realmente puede ser visualizada crece continuamente. Nuestro grupo trabaja en una técnica explorativa para valores continuos que realiza una segmentación en intervalos, para cada uno de los atributos de una consulta, de los valores de todos los registros involucrados en la misma.

5.2 Descripción

Mediante sistemas Data-Warehouse se pueden integrar en una única consulta las múltiples y heterogéneas fuentes de datos que proporcionan todas las compras en cada punto de venta durante un intervalo temporal, sin embargo, ¿es posible y útil representar todos los registros con todos sus campos en una única imagen?, ¿sería el usuario experto capaz de obtener conclusiones y tomar decisiones a partir de tal imagen? Parece lógico pensar que una representación basada en una transformación de

los datos de entrada o en una representación basada en patrones obtenidas por un algoritmo de aprendizaje puede ser mucho más útil y comprensible.

Nuestra propuesta puede verse como una combinación de ambas aproximaciones al integrar un mecanismo de segmentación que visualiza los intervalos más significativos de los atributos más relevantes. Dichos intervalos pueden ser considerados a su vez como reglas de decisión con una única conjunción, de la forma: *Si para el campo A los valores están comprendidos en el intervalo I, entonces P*; donde *P* es una hipótesis de usuario derivada de la visualización del espacio de búsqueda. Para llevar a cabo tal representación, el algoritmo global queda dividido en tres etapas:

1 Para cada atributo A_j involucrado en la consulta se ordenan sus valores y se genera un conjunto inicial de intervalos CI_j (un conjunto por cada dimensión) que representa las similitudes entre las distintas etiquetas dentro de una dimensión. Cada elemento representa a un intervalo en A_j y un histograma con el número de ejemplos cubiertos para cada una de las etiquetas.

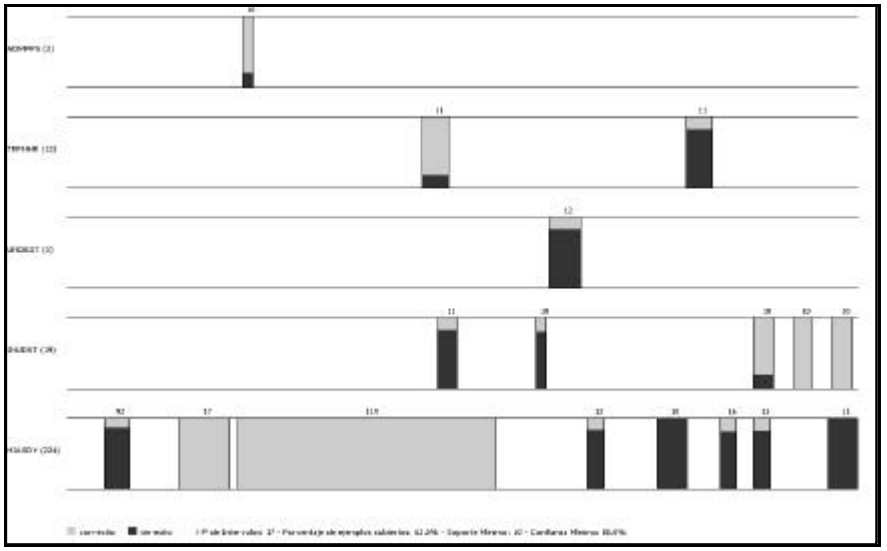

Fig. 2. 62% de los ejemplos cubiertos (de un total de 419) mediante 17 intervalos de confianza mínima del 80% y soporte mínimo 10 perteneciente a 5 atributos de los 6 de la base de datos.

2 A partir de tres parámetros de usuario (cobertura, soporte y confianza) se calcula el que hemos denominado *Conjunto Mínimo de Intervalos Representativos, CMIR*. Para ello, se dispone de un proceso iterativo a partir de una discretización de los atributos intentando maximizar el soporte a partir de una confianza mínima.

3 Por último se visualiza dicho conjunto ***CMIR***, de forma que algunas dimensiones pueden no aparecer al carecer de intervalos significativos. Para representar los distintos intervalos obtenidos en la fase anterior se ha elegido un gráfico de barras horizontales (***CIMR-B***). Cada una de las etiquetas tiene asociada un color

distinto de forma que los intervalos puros (de una sola etiqueta) significativos ocupan un área rectangular de un único color en el interior de la barra que representa al atributo asociado. Para los intervalos impuros, los rectángulos se componen de rectángulos más pequeños de diferente color cada uno cuyo tamaño indica la proporción de ejemplos de cada etiqueta en el interior del mismo.

6. Aplicaciones

6.1 Predicción en series temporales

Mediante un proyecto en colaboración con el departamento de Ingeniería Eléctrica de la U. de Sevilla, se están estudiando técnicas de predicción de Series Temporales. En concreto a los precios de la energía eléctrica en el mercado español, que debido a su reciente liberación son bastante recientes y poco usuales en la literatura actual. Hasta ahora, los métodos más usados para la predicción de los precios de la energía en un mercado competitivo son los distintos tipos de Redes Neuronales Artificiales [16]. Actualmente nuestra línea consiste en la aplicación de algoritmos basados en los vecinos más cercanos Se ha analizado de forma teórica cual es el mínimo error que se comete en la predicción de una serie temporal con este método basado en un algoritmo de búsqueda de los k vecinos más cercanos y se están estudiando las distintas variantes de esta técnica que pueden dar lugar a mejores predicciones [17], variantes que surgen principalmente de las diferentes métricas que pueden ser elegidas así como el número de vecinos. Entre las métricas usadas podemos destacar por los resultados obtenidos la distancia euclídea ponderada por unos pesos estimados mediante un algoritmo evolutivo.

Por otra parte, en el campo de la optimización se están investigando técnicas de búsqueda global de mínimos [18] aplicadas a optimización de sistemas de transporte de energía eléctrica.

6.2 Bioinformática

Actualmente, dentro del campo de la bioinformática, estamos abordando dos problemas específicos: la extracción de genes relevantes en la identificación de subtipos de linfoma y la segmentación intracromosómica de la levadura.

Dentro de la primera línea, estamos investigando la utilidad de algunas técnicas de extracción de atributos relevantes para la selección de los genes que pudieren tener mayor importancia para la posterior clasificación de pacientes con linfoma dentro del tipo Diffuse Large B-cell lymphoma, concretamente entre Germinal Centre B-cell lymphoma y Activated Like B-cell lymphoma. Se han analizado métodos basados en proximidad o vecinos más cercanos, como el algoritmo Relief; otros basados en entropía, como GainInfo; y por último, otros basados en significación estadística, como Chi2. A partir de estas selecciones de genes se generan modelos de conocimiento, en particu-

lar árboles de decisión, que son contrastados con validación cruzada. Además, estamos desarrollando técnicas de clustering específicas para problemas en donde la información es supervisada.

Respecto a la segunda línea, señalar que el mapa genético de cualquier ser vivo está constituido por un conjunto de cromosomas, cada uno de los cuales está a su vez formado por una sucesión consecutiva de genes. Para los investigadores en este campo, en nuestro caso el departamento de Genética de la U. de Sevilla , la segmentación de un cromosoma en un conjunto de intervalos de genes que ocupan posiciones físicamente consecutivas en el cromosoma, tiene interés para descubrir relaciones ocultas entre esos genes o trozos de cromosomas con funciones diferenciadas. Desde el punto de vista del investigador en Minería de Datos, se trata de miles de datos (los genes) cada uno de los cuales viene representado por un conjunto de características dadas por valores numéricos reales.

Para obtener la segmentación solicitada no es posible aplicar técnicas tradicionales de clustering pues los segmentos (o clusters) buscados deben estar formados por genes que ocupen posiciones consecutivas en cada cromosoma. Hay que tener en cuenta que debe tratarse el mapa genómico completo pero a su vez los cluster deben ser obtenidos dentro de cada cromosoma. Nuestra propuesta actual es utilizar un algoritmo evolutivo cuyos individuos representen segmentos de cromosomas mediante los puntos de corte. Los puntos de ruptura conocidos de antemano (por ejemplo, el cambio de un cromosoma al siguiente) son introducidos en la población inicial de forma que no pueden ser modificados a lo largo del proceso evolutivo. El reto principal es encontrar una función de bondad que implemente el objetivo de los investigadores en genética.

6.3 Procesos de desarrollo de software

El desarrollo de software con unos parámetros de tiempo, coste y calidad adecuados es un proceso de producción atípico por las características especiales del producto que se obtiene. Sin embargo, es posible la creación de bases de datos bien reales bien mediante simulación en las que cada registro guarda la información de alguna unidad de software (función, módulo, programa o aplicación) en cuanto a su complejidad: puntos de función, bucles o medida similar o en cuanto a las características de su desarrollo, experiencia o dedicación de los técnicos, esfuerzo y coste o calidad.
Nuestro proyecto CICYT TIC-1143-C03-02 es un proyecto coordinado que cuenta con nuestras herramientas para la obtención de información útil mediante reglas de distinta índole (de decisión, de asociación, cualitativas, gráficas, etc) que permitan estimaciones sobre el proceso de producción de software [19].

7. Referencias

1. Riquelme J y Toro M., “Metodología para obtener nuevas características en sistemas de aprendizaje”, Actas de la VI Conf. de la Asoc. Esp. de IA (CAEPIA '95), pp. 13-24, 1995.

2. Aguilar J, Riquelme J y Toro M, "A tool to obtain a hierarchical qualitative rules from quantitative data", Lecture Notes in Artif Intell. Vol. 1415, pp. 336-346, 1998.
3. Riquelme J, Aguilar J y Toro M.,"Discovering hierarchical decision rules with evolutive algorithms in supervised learning", International Journal of Computer, Systems and Signals, Vo.1 nº 1, pp. 73 - 84, 2000.
4. Aguilar-Ruiz JS, Riquelme J y Toro M, "Data set editing by ordered projection", Intelligent Data Analysis, Vol. 5 nº 5, pp. 405 - 417 IOS Press, 2001.
5. Riquelme JC, Aguilar-Ruiz J y Toro M, "Finding representative patterns with ordered projections", Pattern Recognition, Vol. en prensa, Elsevier Science, 2003
6. Giráldez, R, JS Aguilar-Ruiz, JC Riquelme, FJ Ferrer-Troyano, "Discretization oriented to decision rules generation", Proc. of VI International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2002), pp. 275-279, IOS Press, 2002.
7. Aguilar J, Ruiz R, Riquelme J.C. y Giráldez R, "SNN: a supervised clustering algorithm", Lecture Notes in Artificial Intelligence. Vol. 2070, pp. 207–216, Springer-Verlag, 2001.
8. Mata J, Alvarez JL y Riquelme JC, "Discovering Numeric Association Rules via Evolutionary Algorithm", Lecture Notes in Artificial Intelligence. Vol. 2336, pp. 40 – 51, 2002.
9. Ferrer FJ, JS Aguilar, JC Riquelme, "Non-Parametric Nearest Neighbour with Local Adaptation". Lecture Notes in Artificial Intelligence. Vol. 2258, pp. 22–29, 2001
10. Aguilar J, Riquelme J y Toro M, "Decision queue classifier for supervised learning using rotated hyperboxes", Lecture Notes in Artificial Intelligence. Vol. 1484, pp. 326–336, 1998
11. Riquelme J.C., Giráldez R, Aguilar J y Ruiz R, "Separation surfaces through genetic programming", Lecture Notes in Artificial Intelliigence Vol. 2070, pp. 428 – 433, 2001.
12. Alvarez JL, Mata J y Riquelme J, "OBLIC: Classification System using Evolutionary Algorithm", Lecture Notes in Computer Science. Vol. 2085, pp. 644 – 651, 2001.
13. Aguilar-Ruiz JS, Riquelme JC, Toro M, "Evolutionary learning of hierarchical decision rules", IEEE Systems, Man and Cibernetics Part B, en prensa, 2003.
14. Aguilar-Ruiz JS, JC Riquelme, C. del Valle, "Improving the Evolutionary Coding for Machine Learning Task", Pro. of 16th European Conference on Artificial Intelligence (ECAI 2002), pp. 173–177,IOS Press, 2002
15. Ruiz R, JS Aguilar-Ruiz, JC Riquelme, "SOAP: efficient feature selection of numeric attributes" Lecture Notes in Artificial Intelligence, Vol. 2527, pp. 40 – 51, 2002
16. Troncoso A, JM Riquelme, JC Riquelme, A. Gómez, JL Martínez, "A Comparison of Two Techniques for Next-Day Electricity Price Forecasting". Lecture Notes in Computer Science. Vol. 2412. pp. 384-390, 2002.
17. Troncoso A, JC Riquelme, JM Riquelme, JL Martínez A. Gómez, , "Electricity Market Price Forecasting: Neural Networks versus Weighted-Distance k Nearest Neighbours". Lecture Notes in Computer Science. Vol. 2453. pp. 321-331, 2002.
18. Riquelme JM, A. Troncoso, A. Gómez, J. L. Martínez "Finding Improved Local Minima of Power System Optimization Problems by Interior-Point Methods. IEEE Trans. on Power Systems, en prensa, 2003.
19. Aguilar-Ruiz JS, Ramos I, Riquelme JC y Toro M, "An evolutionary approach to estimating software development projects", Information and Software Technology, Vol. 43(14), pp. 875 – 882 Elsevier Science, 2001.