Búsqueda de Patrones Emergentes Extendidos en Problemas con Datos Mezclados e Incompletos

Milton García-Borroto, J. Francisco Martínez-Trinidad,
José Ruiz-Shulcloper

Reporte Técnico No. CCC-08-002
27 de Febrero de 2008

Coordinación de Ciencias Computacionales
INAOE

Luis Enrique Erro 1
Sta. Ma. Tonantzintla,
72840, Puebla, México.

Búsqueda de Patrones Emergentes Extendidos en Problemas con Datos Mezclados e Incompletos

Milton García-Borroto, J. Francisco Martínez-Trinidad,
José Ruiz-Shulcloper

Coordinación de Ciencias Computacionales
Instituto Nacional de Astrofísica, Óptica y Electrónica
Luis Enrique Erro # 1, Santa María Tonantzintla, Puebla, 72840, México
{milton,ariel,fmartine}@inaoep.mx

Resumen. En esta propuesta de tesis doctoral se aborda el problema de la búsqueda automática de patrones emergentes extendidos en problemas con datos mezclados e incompletos. Se propone como resultado preliminar un algoritmo basado en la inducción de árboles de decisión, para la extracción posterior de los patrones. Los resultados obtenidos en una comparación con algunos de los clasificadores más eficaces y populares muestran que el clasificador construido con estos patrones posee un buen desempeño.

Palabras Clave. Extracción automática de patrones, patrones emergentes, clasificación supervisada, datos mezclados e incompletos

1. Introducción

La clasificación supervisada ha sido abordada desde diferentes enfoques en las áreas de Reconocimiento de Patrones, Inteligencia Artificial y Minería de Datos. Entre los clasificadores más populares se encuentran los estadísticos [1], las redes neuronales[2], vecinos más cercanos [3], árboles de decisión [4] y basados en reglas [5].

Un enfoque diferente a los anteriores se basa en el concepto de patrón. Un patrón puede ser definido como una expresión en algún lenguaje que describe a un subconjunto de los datos [6]. Un tipo importante de patrón es aquél capaz de diferenciar los objetos de una clase, y éstos son los utilizados como base para crear clasificadores supervisados.

El primer clasificador basado en patrones fue KORA-3 [7], que utilizaba un tipo de patrón conocido como Rasgo Complejo (RC). Estos patrones son subconjuntos de tres atributos con sus respectivos valores que aparecen suficientemente en una clase, y no aparecen en la otra. No obstante las restricciones del modelo, que sólo permite patrones de 3 atributos Booleanos, dos clases disjuntas y no propone ningún algoritmo eficiente para buscar los RC, KORA-3 fue utilizado con éxito en la solución de problemas geológicos [8]. Posteriormente se realizaron varias extensiones para eliminar algunas limitantes [9], aunque los RC se extraen exhaustivamente utilizando la definición.

Otro tipo de patrón, llamados Patrones Emergentes (PE), fue introducido posteriormente por Dong et.al. en 1999 [10]. Un PE es un conjunto de pares atributo–valor que describe suficientes objetos de una clase, y a pocos objetos del resto de las clases. A diferencia de los clasificadores basados en KORA se ha trabajado extensivamente en la búsqueda de algoritmos eficientes para extraer los EP, ya sea al momento de clasificar [11] como previamente a la clasificación [12]. Desafortunadamente las soluciones encontradas han limitado el lenguaje de representación de los EP, y muchos algoritmos incluyen una discretización *a-priori* de todos los atributos numéricos.

Recientemente han aparecido comparaciones experimentales [13, 14] que incluyen clasificadores basados en patrones emergentes como DeEPs [15] y BCEP [16]. Estos clasificadores obtienen muy buenos resultados, comparables y muchas veces superiores a los clasificadores utilizados en las comparaciones.

Por otra parte en este trabajo abordaremos especialmente el conjunto de problemas en que los objetos están descritos simultáneamente por atributos numéricos y no numéricos, incluyendo también objetos en los que el valor de algunos de sus atributos no se conoce, es decir, hay ausencia de información. En estos casos hablamos de problemas con Datos Mezclados e Incompletos (DMI).

El trabajo con DMI supone usualmente un reto para muchos clasificadores, desarrollándose muchas veces extensiones o modificaciones para poder utilizarlos. En general se pueden distinguir varias estrategias para trabajar con datos mezclados, entre las que se encuentran: discretizar los valores numéricos, considerar los valores no numéricos como si lo fueran, aplicar clasificadores diferentes para cada tipo de atributo e integrar el resultado, ignorar los valores de un tipo de atributo y crear clasificadores especialmente diseñados para ellos. Por otra parte las principales estrategias para tratar la ausencia de información son la estimación de los valores faltantes y la eliminación de objetos y/o atributos con este tipo de valores. En este trabajo no utilizaremos ningún tipo de transformación en los datos por considerar que pueden implicar una pérdida de información, y trabajaremos con los DMI en su forma original.

1.1. Problema a resolver

Extender el concepto de patrones emergentes para dotarlo de una mayor capacidad expresiva que permita enunciar propiedades más generales que los patrones emergentes actuales para alcanzar mayor eficacia de clasificación. Encontrar dicho conjunto de patrones emergentes extendidos en un conjunto de entrenamiento con DMI.

2. Trabajos relacionados

2.1. Rasgos complejos y KORA

El modelo de clasificación KORA-3 fue introducido por el grupo de Bongard [7, 8] en los años 60 en la antigua Unión Soviética. Éste fue el primer algoritmo utilizado para resolver problemas de clasificación supervisada en Geología [17]. Este método consiste en utilizar propiedades (llamadas Rasgos Complejos, RC) que permiten diferenciar a los objetos de dos clases. Estas propiedades son conjuntos de 3 atributos y sus respectivos valores que aparecen frecuentemente en una clase y no aparecen en la otra. Un objeto se considera *caracterizado* por un RC si contiene los valores definidos en el RC en los atributos correspondientes.

De manera general para clasificar con KORA-3 se buscan iterativamente todos los rasgos complejos del conjunto de entrenamiento. Después, para clasificar un objeto se buscan todos los rasgos complejos que lo caracterizan, y cada uno emite un voto según la clase que describe. Finalmente la clase con mayor cantidad de votos es la asignada al objeto.

El algoritmo KORA-3 original presenta varias limitaciones importantes. En primer lugar sólo utiliza rasgos complejos de tamaño 3, por lo que es incapaz de utilizar propiedades útiles con más o menos atributos. Adicionalmente sólo trabaja con atributos Booleanos y problemas con dos clases disjuntas.

En 1998 se introduce una generalización del KORA-3 denominada Fuzzy KORA-Ω [9]. Las principales características de este algoritmo son:

Utiliza RC que caractericen objetos de otras clases, siempre que estos no sobrepasen cierto umbral predeterminado, permitiendo que el algoritmo sea menos sensible al ruido que puede existir en la muestra de entrenamiento.

Permite trabajar con problemas de más de dos clases, incluyendo estructuraciones no disjuntas y difusas.

Pondera los rasgos complejos basándose en el peso informacional de las variables involucradas y la cantidad de objetos que soportan.

Utiliza criterios de comparación Booleanos, en lugar de atributos Booleanos, permitiendo trabajar con DMI.

Permite trabajar con distintas funciones de semejanza, no sólo la igualdad componente a componente.

Se pueden utilizar rasgos complejos de cualquier longitud, no sólo 3 como en el KORA original.

No obstante sus ventajas sobre el algoritmo original, no se incluye ningún algoritmo para encontrar los rasgos complejos, y éstos son buscados exhaustivamente. La búsqueda se realiza en subconjuntos de atributos previamente seleccionados, pero no se propone una manera de encontrarlos automáticamente, siendo éstos definidos por el usuario.

2.2. Patrones Emergentes

El término patrón emergente (EP, por sus siglas en inglés) fue introducido por primera vez por Dong y Li en 1999 [10]. En este trabajo un EP es definido como un conjunto de pares atributo-valor que aparecen una cantidad de veces significativamente mayor en una clase que en el resto de las clases de un problema dado. Se considera que un objeto *es soportado* por un EP si tiene los valores del EP en sus respectivos atributos.

Con el objetivo de simplificar la notación en este documento describiremos a los objetos como el conjunto de sus pares atributo–valor (componentes), en vez de utilizar una t-upla de valores o la notación matricial. De manera análoga consideraremos un *patrón* como un conjunto de componentes. De esta forma podemos definir formalmente que un patrón X soporta a un objeto O si $X \subseteq O$, es decir, si los componentes de X están *todos* incluidos en O.

Usualmente la propiedad de "significativamente superior" se mide con un umbral que indica cuál es el soporte mínimo necesario para que un patrón sea considerado un EP, siendo el *soporte* de un patrón X en una clase C la fracción de objetos de dicha clase soportados por X, es decir:

$$supp_C(X) = \frac{count_C(X)}{|C|}$$

Donde $count_C(X) = |\{O \in C : X \subseteq O\}|$ es el número de objetos de la clase C soportados por el patrón emergente X. Por otra parte para medir la relación entre los soportes de un EP en dos clases C_1 y C_2 se introduce el concepto de *tasa de crecimiento*, definido por:

$$GrowthRate(X) = \begin{cases} 0 & \text{si } supp_{C_1}(X) = 0 \wedge supp_{C_2}(X) = 0 \\ \infty & \text{si } supp_{C_1}(X) = 0 \wedge supp_{C_2}(X) > 0 \\ \dfrac{supp_{C_2}(X)}{supp_{C_1}(X)} & \text{en otro caso} \end{cases}$$

Podemos decir entonces que X es un patrón emergente, si tiene una tasa de crecimiento mayor a un cierto umbral ξ. Por otra parte un EP con tasa de crecimiento ∞ lo llamaremos *EP de Salto* (Jumping Emerging Pattern, JEP). Estos EP han sido los más utilizados [12, 18] por su capacidad de diferenciar entre clases, aunque son en general muy sensibles al ruido. Es por esto que se han generado algunas extensiones tolerantes al ruido (Noise-tolerant EP, NEP y Generalizad NEP, GNEP [12]). Otros patrones emergentes que han sido utilizados [18] son los JEP fuertes (Strong JEP, SJEP), que son JEP minimales, lo que significa que no contienen ningún otro JEP.

Otra medida utilizada en varios trabajos para evaluar la calidad de un EP es su fuerza entre dos clases C_1 y C_2, la que se define como:

$$strength(X) = \frac{GrowthRate(X)}{GrowthRate(X)+1} \cdot supp_{C_2}(X) = \frac{supp_{C_2}(X) \cdot supp_{C_2}(X)}{supp_{C_2}(X) + supp_{C_1}(X)}$$

Como puede notarse el concepto de Patrón Emergente es muy similar a la extensión del concepto de Rasgo Complejo[9].

En [10] se considera que extraer los EP automáticamente de una muestra de entrenamiento es un problema complicado porque:

No se cumple la propiedad "Apriori" [19], ya que un subconjunto de un EP puede no ser EP. Esto se debe a que si bien su soporte no puede ser menor que el original, si puede tener un soporte mayor que el permitido en alguna otra clase.

Para bases de datos de alta dimensionalidad y un umbral pequeño pueden existir gran cantidad de EP.

Los algoritmos de búsqueda exhaustiva pueden tardar mucho tiempo.

Para resolver estos problemas los autores proponen buscar un subconjunto particular de EP denominados EP frontera. Los EP frontera son tanto los minimales como los maximales de la relación de inclusión entre todos los EP. Por ejemplo, si tenemos los componentes a, b, c, d, e y los EP $E_1 = \{a,b\}$, $E_2 = \{a,b,c\}$, $E_3 = \{a,c\}$ y $E_4 = \{a,b,c,d,e\}$, entonces E_1 y E_3 son minimales ya que no contienen tienen ningún otro EP, mientras que E_4 es maximal ya que no está contenido en ningún otro EP. De esta forma E_1, E_3 y E_4 constituyen los elementos de la frontera. En el trabajo de Dong y Li [10] se proponen algoritmos eficientes para encontrar la frontera y se muestra con algunas bases de datos de prueba que el proceso de búsqueda de los EP se puede realizar rápidamente.

2.2.1. Métodos que buscan patrones al clasificar

En el 2000 Li et.al. presentaron un clasificador basado en EP [11], denominado DeEP (Decisión basada en EP). Este algoritmo no extrae todos los EP de la muestra de entrenamiento, sino que busca aquellos patrones contenidos en el objeto que se desea clasificar. Esta estrategia posee algunas ventajas como que elimina la fase de búsqueda inicial de EP que puede ser tardada, permite que el algoritmo trabaje con matrices cuyos objetos cambien sin tener que ser re-entrenado y no necesita de una discretización *a priori* de los atributos numéricos. Sin embargo esto hace más lenta la etapa de clasificación y no se pueden aplicar algoritmos de selección de los mejores EP. Además se tiene que seleccionar un parámetro denominado factor de vecindad (α) para la comparación entre atributos numéricos, considerándose dos valores x_1 y x_2 como semejantes si $x_1 \in [x_2 - \alpha, x_2 + \alpha]$.

El algoritmo puede describirse de la siguiente forma:

Dado el objeto a clasificar q eliminar del conjunto de entrenamiento todos los componentes que no son elementos de q.

Buscar los EP frontera, utilizando un algoritmo eficiente.

Seleccionar los SJEP de la frontera, que son los que se utilizan en la clasificación.

Asignar a q la clase mayoritaria entre los EP seleccionados.

La experimentación realizada muestra que el clasificador obtenido es mejor en la mayoría de las 21 bases de datos de prueba utilizadas que otros 5 clasificadores seleccionados: C4.5[4], Naive Bayes [20], TAN [21], CBA [22] y LB [23].

Una versión ampliada de este trabajo apareció en 2004 [15]. En ella se incluye un estudio sobre la influencia del factor de vecindad (α) en la calidad del resultado, así como un algoritmo para calcularlo automáticamente. Además se definen dos medidas de evaluación de los patrones encontrados, una basada en la frecuencia o soporte, y la otra en la longitud de los patrones. También se propone la combinación de DeEPs con un clasificador de vecinos más cercanos (NN), nombrándose el clasificador resultante DeEPsNN. Este clasificador utiliza una vecindad del objeto a clasificar definida por un conjunto de parámetros α_i dados por el usuario. Si en esta vecindad se encuentra algún objeto de la muestra de aprendizaje, se aplica un NN tradicional. De no ser así, se aplica DeEPs con todo el conjunto de objetos de entrenamiento. En una comparación con 30 bases de datos de prueba DeEPsNN superó a C5.0 y 3-NN en la mayoría de los casos, aunque no se realizaron pruebas de significación estadística. En [24] se expone una forma de seleccionar automáticamente los α_i, buscando el valor con el que se obtiene la máxima eficacia, tomando el 10% de la muestra de entrenamiento como control para esta estimación.

Como el costo computacional de encontrar los patrones con DeEPs en bases de datos grandes puede ser excesivo, Inakoshi y Ando et.al. [25] propusieron DeEPsLB. Este clasificador realiza la búsqueda de patrones entre los k vecinos más cercanos (utilizando la distancia City-block, también conocida como distancia de Manhattan) del objeto a clasificar, en lugar de utilizar toda la muestra de entrenamiento. El valor de k es estimado mediante validación cruzada con los elementos de la muestra de entrenamiento, seleccionando aquél con el que se obtiene un menor error de clasificación. Los resultados

experimentales mostraron que el clasificador obtenido es mucho más rápido, y que sus resultados son similares a los del DeEPs original.

En un trabajo de Zhang en 2000 [26] se proponen restricciones externas e internas para acelerar el proceso de búsqueda de los EP. Las restricciones externas son umbrales definidos por el usuario para algunos parámetros, como el soporte mínimo y la tasa de crecimiento. Las restricciones internas se imponen con respecto a parámetros calculados por el propio algoritmo, como el soporte del subconjunto de patrones encontrados. Todas estas restricciones permiten detener anticipadamente la búsqueda de EP, con lo que se acelera su extracción y se genera una menor cantidad de los mismos, acelerando el clasificador.

2.2.2. Métodos de búsqueda previa de Patrones Emergentes antes de clasificar

Los métodos de búsqueda de patrones emergentes antes de la clasificación fueron introducidos por Fan y Ramamohanarao en 2002 [18]. A diferencia de los métodos anteriores, se busca inicialmente todos los patrones del conjunto de entrenamiento, los que posteriormente se utilizan para clasificar (Como se hace en [7-9]). La búsqueda inicial permite que los patrones no tengan que ser buscados cada vez que se clasifique un objeto, por lo que acelera notablemente el proceso de clasificación, aunque se necesita un tiempo muy superior al de los algoritmos anteriores para el entrenamiento. Además, se pierde la propiedad de poder trabajar con matrices que cambien dinámicamente.

Un algoritmo general para este tipo de métodos es como sigue:

Procesamiento previo, entrenamiento:

Definir los valores de los parámetros del algoritmo.

Discretizar los atributos numéricos.

Representar los objetos con una estructura de árbol.

Encontrar todos los patrones emergentes con algún tipo de recorrido del árbol. Usualmente en esta etapa hay que agregar algún tipo de procesamiento extra al recorrer algunos nodos.

Post-procesamiento de los EP encontrados.

Clasificación:

Para clasificar un objeto O, dado un conjunto de patrones emergentes por clase.

Para todas la clases del problema

Encontrar los EP que soportan al objeto O.

Sumar el voto de cada patrón emergente encontrado utilizando alguna función basada en el soporte y la tasa de crecimiento de cada patrón.

Asignar al objeto la clase con mayor voto total.

En el trabajo de Fan y Kotagirij [18] para lograr la búsqueda exhaustiva en un tiempo razonable se buscan los SJEP, que son usualmente mucho menos que los JEP. Por ejemplo en la base de datos Waveform del repositorio de la UCI [27] se pueden encontrar 4´096´477

JEP, contra sólo 7939 SJEP. Todos los atributos numéricos son discretizados *a priori*, por lo que el problema se resume a encontrar los subconjuntos de componentes que son SJEP.

Para realizar eficazmente la búsqueda se representa toda la información del conjunto de entrenamiento en forma de árbol, que se conoce como Árbol de Patrones de Contraste. Este árbol es un árbol de múltiples vías ordenadas donde cada nodo posee un conjunto variable de hijos, cada uno asociado a un componente. Para ordenar los ítems de un nodo se utiliza una relación de orden total ($\prec$) que refleja la importancia del item para clasificar (utilizando la tasa de crecimiento). Como ejemplo [12], supongamos que tenemos un conjunto de objetos de dos clases con los siguientes componentes ordenados: $\mathrm{e} \prec \mathrm{a} \prec \mathrm{b} \prec \mathrm{c} \prec \mathrm{d}$ (Tabla 1). El árbol de contraste asociado se muestra en la Figura 1.

Tabla 1. Ejemplo de un conjunto de objetos representados por componentes y distribuidos en dos clases

ID	Clase	Componentes en el objeto	Componentes ordenados
1	C_1	$\{a,c,d,e\}$	$\{e,a,c,d\}$
2	C_1	$\{a\}$	$\{a\}$
3	C_1	$\{b,e\}$	$\{e,b\}$
4	C_1	$\{b,c,d,e\}$	$\{e,b,c,d\}$
5	C_2	$\{a,b\}$	$\{a,b\}$
6	C_2	$\{c,e\}$	$\{e,c\}$
7	C_2	$\{a,b,c,d\}$	$\{a,b,c,d\}$
8	C_2	$\{d,e\}$	$\{e,d\}$

Es importante notar que en el árbol de contraste cada objeto está representado por un camino entre el nodo raíz y una hoja en dependencia de sus componentes. Por ejemplo, el objeto con $ID = 1$ con componentes $\{e,a,c,d\}$ está representado por el camino punteado entre la raíz y el nodo resaltado **A** en la Figura 1. De la misma forma los nodos resaltados **B** y **C** representan los objetos $\{e,b\}$ y $\{a,b,c,d\}$ respectivamente. Cada nodo almacena, además del componente asociado, la distribución de las clases de los objetos que representa. Esta información es la que aparece en la parte inferior del nodo, primero el número de objetos soportados de la clase C_1 y luego el número de objetos soportados de la clase C_2.

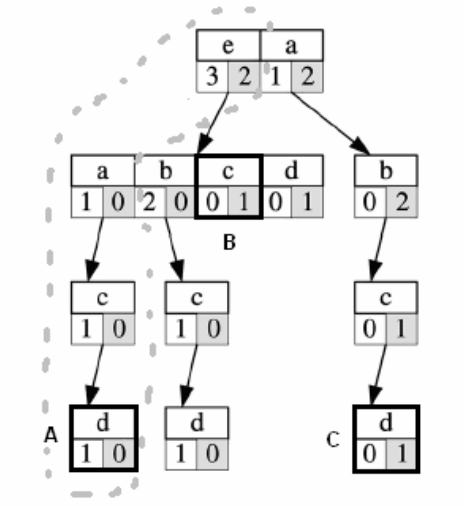

Figura 1. Árbol de contraste que representa los objetos de la Tabla 1.

El algoritmo de construcción del árbol es sencillo y consiste en ir incluyendo la información de cada objeto uno por uno. Los objetos son previamente ordenados según los componentes que contengan, y el orden de cada uno de dichos componentes. Para cada objeto O se ejecuta el siguiente algoritmo:

1. $NodoActual \leftarrow Raiz$
2. Si O tiene algún componente de los representados en $NodoActual$
   2.1 Buscar el nodo hijo asociado $NodoActual.HijoAsociado$ y hacer $NodoActual \leftarrow NodoActual.HijoAsociado$. Ir a paso 2

En otro caso

Crear nuevo hijo, asociarlo con el componente de mayor importancia (según $\prec$) aún no encontrado en el recorrido del árbol, hacer $NodoActual \leftarrow NuevoNodo$ e ir a paso 3.

3. Si queda algún componente de O sin representar en el recorrido actual ir a 2, en otro caso terminar

El proceso de creación del árbol para los objetos de la Tabla 1 puede verse en la Figura 2. Nótese que en cada inserción o paso por un nodo se actualizan los contadores por clases, y que el árbol final incluye toda la información del conjunto de entrenamiento original.

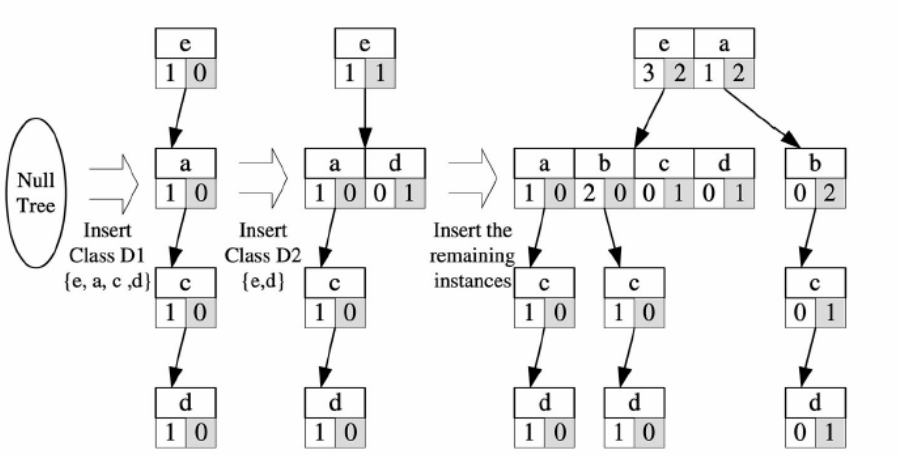

Figura 2. Construcción paso por paso de un árbol de contraste.

En el árbol de contraste se encuentra resumida toda la información necesaria para la búsqueda de los EP. Por ejemplo, observando el árbol de la Figura 1 y suponiendo que buscamos patrones emergentes de soporte mayor o igual que 2 en una clase y 0 en la otra, tenemos lo siguiente:

$\{e,a\}$ no es un EP, ya que no tiene el soporte mínimo. Además, no hay que seguir buscando en sus hijos, pues ninguno de ellos puede alcanzarlo.

$\{e,b\}$ es un JEP, que tiene soporte 2 en la clase C_1 y 0 en la clase C_2, y ninguno de sus hijos puede generar otro patrón emergente de soporte superior. Por tanto si queremos los EP de máximo soporte podemos no explorar sus hijos.

No todos los patrones pueden ser obtenidos por un recorrido del árbol, ya que éste no tiene en cada nodo la información completa. Por ejemplo el patrón $\{e,c\}$ según el árbol tiene soporte (0,1) en las clases C_1 y C_2 respectivamente, pero buscando en la Tabla 1 observamos que el soporte real es (2,1). Esto ocurre así por la forma de construcción del árbol, ya que dicho nodo no representa realmente al patrón $\{e,c\}$, sino a los objetos que tienen los componentes e y c, pero que no tienen ni a ni b. Puede verse en el árbol, por ejemplo los nodos que representan los patrones $\{e,a,c\}$ y $\{e,b,c\}$, y que no están incluidos en los soportes asociados al nodo de $\{e,c\}$.

Para resolver el problema de la representación incompleta de la información se ejecuta un procedimiento de mezcla de árboles en cada nodo [12].

El algoritmo de Fan y Ramamohanarao [18] de clasificación basada en EP consiste en dos etapas, el entrenamiento y la clasificación. En la fase de entrenamiento se encuentran los patrones emergentes que cumplen con los criterios de cobertura dados, construyendo un árbol de contraste y utilizando el algoritmo de búsqueda y mezcla explicado anteriormente. En la fase de clasificación se realizan los siguientes pasos:

Buscar los EP de cada clase que soportan al objeto a clasificar.

Calcular la suma de los soportes de los EP para cada clase.

Normalizar las sumas, dividiéndolas por la cantidad de patrones por clase. Esto se realiza, según el criterio de los autores, para mitigar el efecto de que la cantidad de patrones

encontrados para cada clase puede estar desbalanceada, pudiendo sesgar la clasificación hacía la clase de más patrones.

Asignar al objeto la clase con mayor suma normalizada.

En 2006 apareció una versión extendida de este trabajo [12] en la que se mejoran los algoritmos y se incluyen las modificaciones necesarias para extraer patrones tolerantes al ruido. En este trabajo se muestra una comparación experimental de los algoritmos introducidos con C4.5 [4], NaiveBayes [20], Bagging [28], Boosting [29] y RandomForest [30], utilizando 27 bases de datos de prueba. Según los resultados de la comparación, el uso de los SJEP y sus extensiones permite obtener resultados superiores en la mayoría de las bases de datos.

En 2002 Bailey et.at. reportaron algoritmos nuevos para la búsqueda de los EP, utilizando otra estructura de árbol diferente al árbol de contraste, con un reordenamiento para encontrar rápidamente los patrones más frecuentes. En este trabajo no se encuentran todos los JEP, sino un subconjunto de los más importantes, utilizando umbrales. Según los autores estos algoritmos son aproximadamente 5 veces más rápidos que el original, pero se encontró que en varias bases de datos se degrada significativamente la calidad del clasificador resultante.

Fan y Ramamohanarao en 2003 [16] propusieron un híbrido de EP con un clasificador Bayes simple [20]. Este enfoque mitiga algunos problemas importantes de clasificadores basados en EP anteriores, entre los que se encuentran:

El proceso de normalización de la suma de votos por clase, aunque intuitivo, puede resultar contraproducente.

El uso de JEP es efectivo si hay suficientes en la base de datos, pero en algunas de ellas puede incluso no haber ninguno con un soporte suficientemente alto.

Usualmente existe mucha redundancia en los EP extraídos, lo que puede entorpecer el trabajo del clasificador.

Entre las soluciones aportadas en [16] se encuentran la eliminación de patrones basada en el soporte por clases, lo que significa que un patrón tiene que soportar al menos a un objeto que el resto de los patrones no soporten. Para esto se ordenan los patrones por su soporte y longitud, y después se realiza un procedimiento iterativo, donde cada patrón es agregado al resultado si soporta al menos un objeto nuevo con respecto a los patrones que ya han sido seleccionados antes. Una idea similar a ésta ya estaba incluida en el algoritmo KORA-3 original de Bongard [7].

Finalmente para la clasificación se utiliza la siguiente expresión, que estima la probabilidad que el objeto O pertenezca a la clase C_i:

$$P(O,C_i) = P(C_i)\frac{\prod_{u\in numerator} P(u,C_i)}{\prod_{v\in denominator} P(v,C_i)}$$

Donde $P(C_i)$ es la probabilidad *a priori* de las clases y tanto *numerator* como *denominator* son dos subconjuntos de objetos, el primero con todos los objetos soportados por los patrones que serán utilizados de la clase, y el segundo con todos los objetos que son soportados por más de un patrón. Finalmente se asigna al objeto a clasificar la clase con mayor probabilidad.

En 2004 Alhammady y Ramamohanarao crearon el primer clasificador basado en EP para la clasificación de las clases minoritarias en problemas con clases desbalanceadas [31]. Para este propósito el conjunto de EP es modificado de la siguiente forma:

Generando nuevos EP para la clase minoritaria no descubiertos en el proceso original.

Eliminando EP de la clase mayoritaria que soporten a algún objeto de las clases minoritarias.

Incrementando el soporte de los EP de clase minoritaria.

Un trabajo posterior de los mismos autores [32] combina los EP con árboles de desición para lograr el mismo propósito. En este caso se generan nuevos objetos de las clases minoritarias y se sobre-muestrea los objetos existentes de dichas clases.

En ambos casos se encuentra que los clasificadores resultantes tienen mejores resultados que otros algoritmos como PNRule [33], C4.5 [4], MetaCost [34] y sobre-muestreo [35].

Recientemente se han aplicado los patrones emergentes al descubrimiento de conocimiento en bases de datos relacionales, donde la información está usualmente dispersa en diferentes tablas [36].

3. Motivación

Como puede apreciarse en los epígrafes anteriores los clasificadores basados en patrones, en particular los basados en patrones emergentes, permiten alcanzar eficacias comparables con la de otros clasificadores del estado del arte, incluso en problemas con clases desbalanceadas. Además, en la mayoría de los casos resulta fácil la interpretación de los resultados de la clasificación. No obstante todos presentan algunas de las siguientes deficiencias importantes:

No se tienen algoritmos eficientes de búsqueda de los patrones.

No se buscan todos los patrones en la fase de entrenamiento, sino al clasificar un objeto. No obstante todas las mejoras introducidas a los algoritmos, la clasificación sigue siendo lenta y no se pueden evaluar los EP en su conjunto.

Se ha simplificado el lenguaje de representación de los EP a sólo dos propiedades simples, para lograr extraerlos eficientemente. Esta simplificación incluye la discretización *a priori* de todos los atributos numéricos, con las desventajas que esto conlleva.

No se ha estudiado suficientemente la influencia en el resultado de los datos faltantes.

No se ha estudiado con suficiente profundidad el problema de la abstención al clasificar, pudiendo ocurrir las abstenciones con bastante frecuencia.

Es por esto que consideramos importante generalizar el concepto de patrón emergente dotándolo de mayor capacidad expresiva, con lo cual se puedan construir propiedades más generales. Consideramos que esto puede tener un impacto positivo en la calidad de los clasificadores basados en EP. Se necesitan además algoritmos capaces de encontrar estos EP en conjuntos de entrenamiento con DMI.

4. Propuesta

4.1. Pregunta de Investigación

¿Cómo extraer patrones emergentes de mayor capacidad expresiva de una muestra de entrenamiento con datos mezclados e incompletos, para obtener clasificadores más eficaces que los existentes?

4.2. Objetivo General

Diseñar un método para extraer patrones emergentes de mayor capacidad expresiva, a partir de un conjunto de entrenamiento con datos mezclados e incompletos, para obtener clasificadores más eficaces que los existentes.

4.3. Objetivos particulares

Extender el lenguaje de representación de los patrones emergentes para poder expresar propiedades más generales.

Diseñar un método de extracción de patrones emergentes con el lenguaje extendido en bases de datos con DMI.

Proponer un método de evaluación y filtrado de patrones emergentes que permita su selección y pesado.

Obtención de métodos o heurísticas de estimación de buenos parámetros para los métodos propuestos.

Diseñar de un nuevo clasificador basado en patrones emergentes con eficacia superior a los clasificadores existentes.

4.4. Metodología

Seleccionar, recopilar y analizar críticamente la bibliografía pertinente.

Extender el lenguaje de representación de patrones emergentes utilizado hasta el momento para incluir propiedades como:

$AtributoNoNumérico = v$ o $AtributoNoNumérico \neq v$

$AtributoNumérico \leq v$ o $AtributoNumérico :> v$

$AtributoNoNumérico = v_1$ o $AtributoNoNumérico = v_2$ … o

$AtributoNoNumérico = v_p$

$AtributoNumérico \in [v_1, v_2]$

$AtributoNoNumérico \in \{v_1, v_2, \ldots v_p\}$

Propiedades de este tipo fueron utilizadas en la definición de l-complejo de Michalsky [37] para la construcción de conceptos.

Proponer un nuevo método de extracción de los patrones extendidos en matrices con DMI:

Creación de un nuevo algoritmo de extracción de patrones emergentes extendidos en bases de datos con datos mezclados e incompletos. Para esta tarea se realizará:

Estudio crítico de los métodos de extracción de patrones emergentes existentes.

Proponer un nuevo algoritmo para encontrar EP extendidos.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

Definir métodos o heurísticas para determinar cuándo un conjunto de EP es representativo de una base de datos. Esto puede permitir detener anticipadamente el proceso de búsqueda de EP, ahorrando tiempo de procesamiento. Para este punto se parte

del presupuesto de que existe mucha redundancia en el conjunto de todos los EP, por lo que un subconjunto puede representarlo sin pérdida de información. Para ello se realizará:

Estudio crítico de las medidas de calidad de EP existentes.

Proponer un nuevo método o heurística que permita determinar cuándo un conjunto de EP es representativo de una base de datos.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

Buscar criterios de calidad de los EP encontrados, y aplicarlos al filtrado de EP post-extracción. Existen muchas medidas para evaluar la calidad de un EP [6, 14] que pueden ser utilizadas para el filtrado de los patrones, o para la asignación de un peso a cada uno para clasificar. Por la importancia de este tema, así como el impacto directo que tiene en la calidad de un clasificador basado en EP se realizará:

Análisis crítico de las formas existentes de evaluar la calidad de un EP.

Evaluación del desempeño de cada medida de calidad en la clasificación de bases de datos de prueba. En este paso se utilizará el clasificador propuesto por Fan y Ramamohanarao [12] utilizando el subconjunto de los mejores EP evaluados por cada medida o todos los EP con el peso calculado con dicha medida.

Desarrollo de nuevos criterios de calidad para patrones emergentes, que permitan realizar una selección de los mejores o una asignación de pesos al clasificar.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

d) Buscar una estrategia de cálculo automático de los parámetros de extracción de los EP. El concepto de EP incluye la expresión "soporte significativamente superior" en una clase que en las demás. La forma habitual de expresar numéricamente está propiedad es mediante el uso de umbrales. En la mayoría de los trabajos previos estos umbrales son fijos o definidos por el usuario. En este trabajo buscaremos un algoritmo adaptativo de búsqueda inicial y refinamiento de estos umbrales. Para ello se realizará:

Estudio de las características de un problema de clasificación específico que pueden ser útiles para la estimación de valores de los umbrales cercanos a los deseados.

Construcción de un algoritmo adaptativo de refinamiento de los valores de los umbrales.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

Definir estrategias para clasificar los objetos en los que nuestro clasificador se abstiene. El problema de la abstención al clasificar no ha sido muy estudiado en la literatura de reconocimiento de patrones. La mayoría de los clasificadores definen una estrategia de que hacer si no poseen evidencia que les permita distinguir la clase de un objeto dado. La asignación de la clase mayoritaria o una elección al azar son las más utilizadas. No obstante, en los trabajos de clasificación con patrones emergentes no se ha abordado este problema, teniendo estos clasificadores según nuestros experimentos mayor tendencia a abstenerse.

En nuestro caso buscaremos más evidencia que permita la clasificación. Para esto realizaremos las siguientes tareas:

Estudio de los tipos de abstención que comete el clasificador propuesto, así como las causas de cada una de ellas.

Búsqueda de estrategias para lograr la clasificación para cada tipo de abstención.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

f) Trabajo con datos faltantes. Un dato faltante es un valor de un atributo en un objeto determinado que, o bien no se conoce, o no se pudo medir, o no tiene sentido para un objeto en particular. La presencia de este tipo de datos en un problema presupone un reto importante para muchos algoritmos de clasificación. Es por eso que en nuestro trabajo nos proponemos realizar:

Estudio crítico del trabajo con datos incompletos para la clasificación supervisada.

Selección del esquema más adecuado para la tarea de extraer EP.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

Proponer un nuevo clasificador basado en EP. Dado que no existen métricas desarrolladas para evaluar la calidad de un conjunto de patrones emergentes, utilizaremos como medida de calidad la eficacia de un clasificador basado en ellos. No obstante también trabajaremos en la búsqueda de métricas basadas en propiedades independientes del clasificador. Para ello realizaremos:

Estudio comparativo de los algoritmos de clasificación basados en patrones emergentes.

Selección del algoritmo a utilizar para nuestras comparaciones.

Búsqueda de métricas independientes de un clasificador para la medición de la calidad de un conjunto de patrones emergentes.

Evaluación de resultados mediante la realización de pruebas y comparación de resultados. Retroalimentación.

Comparación experimental del clasificador obtenido basado en los patrones emergentes extraídos, con otros clasificadores del estado del arte, en bases de datos de prueba.

Análisis de los resultados y elaboración de las conclusiones.

4.5. Cronograma

Tareas	Cuatrimestres						
	1	2	3	4	5	6	7
1. Análisis de la literatura							
2. Extensión del lenguaje de EP							
3. Nuevo método de extracción							
a)Algoritmo basado en árboles							
b)Condiciones de parada							
c)Criterios de calidad de EP							
d)Determinación de parámetros							
e)Manejo de abstención							
f)Manejo de datos faltantes							
4. Clasificador basado en EP							
5. Comparación experimental							
6. Escritura de artículos							
7. Redacción de la propuesta							
8. Redacción del documento de tesis							
9. Defensa de la tesis							

5. Resultados preliminares

5.1. Extensión del lenguaje de representación de patrones emergentes

En la literatura se reportan tres tipos de componentes que forman los patrones emergentes:

$AtributoNoNumérico = v$.

$AtributoNumérico \in [v-\alpha, v+\alpha]$, donde α es un valor único para todos los componentes del algoritmo, que puede ser definido por el usuario [11], o encontrado automáticamente por el algoritmo [15]. Este tipo de componentes sólo es utilizado en los algoritmos que buscan los EP al momento de clasificar.

$AtributoNumérico \in [v_1, v_2]$, donde v_1 y v_2 son encontrados al discretizar el atributo numérico antes de iniciar la búsqueda de EP.

Es de resaltar que con los métodos actuales estos componentes no pueden encontrarse todos en una misma descripción, sino sólo los A y B en los métodos que buscan patrones al clasificar [11, 15, 38], y los A y C en los que buscan patrones *a priori* [12, 18].

Para poder construir propiedades más generales que diferencien objetos entre clases, extendimos el lenguaje de representación de los patrones emergentes utilizados hasta el momento basándonos en el concepto de selector de Michalsky [37]. Los nuevos componentes introducidos fueron:

$AtributoNoNumérico \in \{v_1, v_2, \ldots v_p\}$

$AtributoNoNumérico \neq v$

$AtributoNumérico \leq v$

$AtributoNumérico > v$

Además, deseamos obtener patrones emergentes en los que puedan estar presentes simultáneamente todos los tipos de componentes. Los nuevos componentes creados permiten expresar propiedades más generales que los actuales. No obstante seguiremos buscando en el transcurso de la investigación otras propiedades que sean útiles en la clasificación, para incorporarlas al lenguaje de representación creado.

5.2. Nuevo método de extracción de EP extendidos en conjuntos de objetos con datos mezclados e incompletos

5.2.1. Creación de un nuevo algoritmo de extracción de EP

Para poder extraer EP extendidos en conjuntos de entrenamiento con DMI creamos un método basado en árboles de decisión, debido a que pueden expresarse en ellos las propiedades que nos interesan y son rápidos de construir. Para extraer suficientes EP se requieren muchos árboles de decisión diferentes, por lo cual se propuso un algoritmo de inducción de árboles de decisión que permitiera crear árboles diversos al ser ejecutado varias veces.

Este método se ejecuta un número determinado de veces, creándose diferentes árboles. De cada árbol generado son extraídos un conjunto de patrones emergentes (así como su cobertura) recorriendo los caminos entre la raíz y las hojas seleccionadas. El resultado final del método es la unión de todos los patrones emergentes encontrados, después de eliminar duplicados.

El algoritmo de inducción de árboles propuesto en esta investigación, dado un conjunto de objetos T, es el siguiente:

ConstruirArbol(T), devuelve un nodo del árbol de decisión

Verificar las condiciones de terminación.

Para cada atributo a con más de un valor diferente en T

Encontrar la ganancia de información normalizada para cada división posible de a, utilizando los criterios de generación de nodos según el tipo de cada atributo.

Seleccionar $a_{seleccionado}$ aleatoreamente entre las 5 divisiones de mayor ganancia de información.

Crear un nodo de decisión $NewNode$ que divida a T de acuerdo al atributo $a_{seleccionado}$.

Llamar recursivamente a **ConstruirArbol** con cada subconjunto de T obtenido al dividir de acuerdo a $a_{seleccionado}$, y adicionar los nodos resultantes como hijos de $NewNode$.

Las condiciones que se verifican en el paso 1 utilizan los siguientes parámetros: mínimo soporte necesario para que un nodo homogéneo genere un patrón emergente ($MinSupportPureNode$), el soporte mínimo exigido en un nodo no homogéneo ($MinSupportMixedNode$) y el soporte máximo para el resto de las clases en los nodos no homogéneos ($MaxSupportMixedNode$). Además se utiliza el parámetro Booleano $NodoEsPuro$, el que es verdadero si y sólo si el nodo contiene objetos de una sola clase. Estas condiciones son:

El nodo representa un patrón emergente, es decir, la distribución por clases de los objetos que representa en el conjunto de entrenamiento cumple con las condiciones definidas:

$$\begin{pmatrix}(NodoEsPuro)\wedge(Soporte\geq MinSupportPureNode))\vee\\ \left(\begin{matrix}(\sim NodoEsPuro)\wedge(Soporte\geq MinSupportMixedNode)\wedge\\ (SoporteOtherClass\leq MaxSupportMixedNode)\end{matrix}\right)\end{pmatrix}$$

Con esta expresión permitimos extraer patrones emergentes que son JEP como que no lo son. Un EP que no es JEP sólo puede ser extraído si tienen soporte en su propia clase mayor que el exigido para ser JEP y soporte menor que un umbral en las otras clases.

El nodo no puede generar un patrón, porque su soporte es menor que el mínimo exigido.

En el paso 2.1 del algoritmo que proponemos se dividen los objetos de un nodo de todas las siguientes formas, según el tipo de atributo, escogiéndose posteriormente una de ellas para construir el árbol. Entre paréntesis señalamos los tipos de componentes de los patrones que serán extraídos en cada caso.

Atributos no numéricos

Si existen exactamente dos valores, crear un nodo con dos hijos, uno para la propiedad $VariableNoNumérica = Valor_1$ y otro para la propiedad $VariableNoNumérica = Valor_2$ (Componente tipo A).

- Si existen más de dos valores:

Por cada valor crear un nodo con dos hijos, uno con la propiedad $VariableNoNumérica = Valor$ y otro con $VariableNoNumérica \neq Valor$ (Componentes tipo A y E).

Crear un nodo con un hijo por cada valor y la propiedad $VariableNoNumérica = Valor$ (Componente tipo A).

Se crea un nuevo nodo con un hijo por clase y la propiedad $VariableNoNumérica \in \{Valor_1, Valor_2, \ldots Valor_p\}$. Cada nodo representa un conjunto de valores del atributo que aparecen mayoritariamente en la clase asociada al nodo. Finalmente se crea otro nodo para agrupar los valores no asociados a ninguna clase (Componentes tipo D).

Atributos numéricos

Inicialmente se ordenan los objetos de acuerdo al valor del atributo y se determinan los puntos de corte. Un punto de corte es un objeto que posee al menos un antecesor o sucesor de clase diferente según el orden definido, y llamamos valor de corte al valor de su atributo. Por cada valor de corte generamos un nuevo nodo cuyos hijos corresponden a las propiedades $VariableNumérica \leq Valor$ y $VariableNumérica > Valor$ respectivamente (Componentes tipo F y G).

Se buscan los intervalos de valores que contengan varios objetos de una sola clase, y se genera por cada uno un nodo hijo con la propiedad $VariableNumérica \in [Valor_1, Valor_2]$, siendo $Valor_1$ y $Valor_2$ los extremos del intervalo (componentes tipo C).

Para medir la ganancia de información en el paso 2.1 utilizamos una medida de impureza del nodo basada en entropía, la cual tiene la siguiente expresión:

$$i(t) = -\sum_{j=1}^{c} P_j \log(P_j)$$

Donde c es la cantidad de clases y P_j es la proporción de objetos de la clase j en el nodo t. Ésta es una de las medidas más populares para la inducción de árboles, pero no descartamos en el futuro hacer pruebas con otras medidas.

Hay que destacar que en este algoritmo no se consideran estrategias para el aumento de la calidad del árbol generado a costa de la velocidad de generación, tales como la poda. Esto se debe a que no nos interesa el árbol en sí como clasificador, sino que sólo lo utilizamos para poder encontrar los patrones emergentes.

5.2.2. Criterios para medir la calidad de un patrón emergente. Asignación de pesos

Existen varios criterios para evaluar la calidad de un patrón emergente: soporte, tamaño (número de componentes que lo forman), tasa de crecimiento, fuerza y cobertura, entre otros. Además existen diferentes formas de utilizar esta evaluación, ya sea para filtrar el conjunto de patrones, como para asignar diferentes votaciones a cada uno para clasificar.

Después de analizar las ventajas y desventajas de los criterios existentes decidimos introducir un criterio diferente a los utilizados en la literatura para evaluar los patrones encontrados, y determinamos no eliminar ningún patrón, sino asignarles diferentes pesos al clasificar. El nuevo criterio que proponemos asigna a un patrón un peso proporcional a las diferencias entre los objetos soportados por él, y para esto calcula la máxima disimilaridad entre ellos considerando todos sus atributos. La disimilaridad se calcula con la función de disimilaridad para datos mezclados HVDM [39], aunque exploraremos otras funciones de comparación en el futuro.

Esta forma de asignar el voto trata de mitigar las desventajas de utilizar el soporte, en el que se basan casi todas las medidas reportadas previamente. Un EP con un alto soporte es interpretado usualmente como una propiedad muy importante para definir la clase, pero esto sólo es cierto si los objetos que soporta son diferentes entre sí en los atributos que no están incluidos en el EP. La presencia de muchos objetos casi idénticos en una base de datos hace que se encuentren muchos patrones emergentes con un alto grado de soporte, y estos patrones pueden decidir la clasificación de muchos objetos. Este alto soporte puede ser ficticio, ya que puede deberse a sesgos en el muestreo de los datos, y no a que representan propiedades importantes de las clases del problema.

Nuestro criterio asigna de manera general un voto más bajo a los EP que soportan objetos muy similares, independientemente de cuantos sean, que a los EP que soportan objetos diferentes. Hasta el momento este criterio ha dado resultados superiores a los criterios anteriores en las bases de datos utilizadas, aunque éstas no presentan mucho desbalance. Este resultado era esperado ya que el criterio no se basa en el soporte, siendo menos sensible a la cantidad de objetos por clase.

Otro resultado interesante está relacionado con un efecto que encontramos en algunos casos, en los que la adición de nuevos EP degradaba la calidad del clasificador. Un estudio del problema indicó que se debía a la acumulación de votos de patrones menos útiles, que

terminaban por superar a los mejores. Utilizando el nuevo criterio y normalizando los votos se mitigó casi totalmente este efecto.

También se realizaron pruebas de filtrado, eliminando aquellos patrones que tuvieran un voto normalizado menor que cierto umbral. En general se encontró que existen valores de umbral diferentes para cada base de datos que incrementan la eficacia del clasificador, pero no se pudo encontrar ninguna relación de este valor con propiedades de la base de datos. La estimación de un buen umbral para un conjunto de datos dados constituye una línea futura de investigación.

5.2.3. Algoritmo adaptativo de búsqueda de los parámetros de extracción de los EP

Los principales parámetros para la extracción de EP en nuestro algoritmo son, como ya se mencionó anteriormente, el mínimo soporte de la clase mayoritaria en un nodo homogéneo (*MinSupportPureNode*), el soporte mínimo para un nodo no homogéneo (*MinSupportMixedNode*) y el soporte máximo para el resto de las clases en los nodos no homogéneos (*MaxSupportMixedNode*). Estos parámetros tienen un impacto directo en la calidad de los EP, por lo que su estimación resulta muy importante. Además los mejores valores varían mucho de una base datos a otra.

El parámetro *MinSupportMixedNode* se introduce en el algoritmo para permitir la tolerancia al ruido, que está presente comúnmente en los problemas de reconocimiento de patrones. Es por esto que se establece para este parámetro un valor mucho mayor que para *MinSupportPureNode* , para diferenciar el ruido de lo que es información diferente. Finalmente tomamos $MinSupportMixedNode = 2 \cdot MinSupportPureNode$ y $MaxSupportMixedNode = 1$, asumiendo que el ruido aparece usualmente en objetos aislados, y no en grupos de objetos.

Finalmente se debe estimar solamente el valor de *MinSupportPureNode* , lo que se realizó con la siguiente metodología:

Se elige un valor inicial de *MinSupportPureNode* suficientemente alto. Este valor lo tomamos en nuestros experimentos igual a 30, pero en el futuro trabajaremos en una estimación más cercana al valor final.

Se comienzan a construir árboles con el conjunto de entrenamiento.

Si se logra soportar rápidamente todos los objetos de la muestra de aprendizaje, se aumenta el valor de *MinSupportPureNode* y se regresa al paso 2.

Si faltan muchos objetos por ser soportados de la muestra de aprendizaje después de generado varios árboles, se disminuye el valor de *MinSupportPureNode* y se regresa al paso 2.

Hay que resaltar que esta metodología no fue programada, pues como se nota tiene aun expresiones de vago significado como “soportar rápidamente”, “muchos objetos”, sino que fue ejecutado manualmente para seleccionar los valores. No obstante será la base para la creación del algoritmo adaptativo que realizará el proceso automáticamente en versiones posteriores.

5.2.4. Manejo de la abstención del clasificador

Existen principalmente dos motivos por los que un clasificador se abstiene, es decir, no le asigna ninguna clase a un objeto. Éstas son la falta de información para clasificar un objeto y la acumulación de evidencia contradictoria que no permite al clasificador distinguir entre dos o más clases. Aunque muchos clasificadores tienen potencialmente la posibilidad de abstenerse, en la práctica la mayoría decide por la clase mayoritaria del problema o una clase al azar, lográndose la correcta clasificación de algunos objetos. En nuestra opinión esta técnica dificulta la evaluación real del desempeño de un clasificador, sobre todo si las clases no están balanceadas.

Una de las ventajas fundamentales de que un clasificador se abstenga de tomar una decisión es que esto posibilita al usuario o sistema que utiliza el clasificador buscar más evidencia, o utilizar otro clasificador en un esquema de combinación, lo que puede ser más certero que la clasificación al azar o por la clase mayoritaria.

Por todo lo antes expuesto es que nuestro clasificador incluye la abstención en los dos casos siguientes:

El objeto a clasificar no posee ningún patrón emergente. En este caso se puede o bien combinar el clasificador con otro, o buscar patrones de menos cobertura en la muestra de entrenamiento hasta que se acumule suficiente evidencia para la clasificación.

Los votos para al menos dos clases son muy similares. En este caso el clasificador no puede fiablemente decidir qué clase asignar, y se pueden aplicar soluciones similares a la anterior.

Debemos resaltar que ninguna de las posibles soluciones ha sido implementada aún, por lo que es un elemento a tener en cuenta para la interpretación de los resultados experimentales presentados en este documento. Si se opta por clasificar las abstenciones en la clase mayoritaria, muy posiblemente varias de las abstenciones se convertirían en aciertos, pero en nuestro caso todas fueron contadas dentro de los errores del clasificador.

5.2.5. Trabajo con los datos faltantes

Los datos faltantes generan retos importantes para muchos clasificadores, de tal forma que se ha sugerido en algunos trabajos incluso eliminar totalmente los atributos y objetos en los que aparezcan. No obstante, este tipo de medidas drásticas pueden provocar que se desaproveche mucha información útil. Es por esto que se han introducido muchas formas de tratarlos, como la estimación de su valor.

En este trabajo utilizamos para la inducción del árbol con datos faltantes un esquema diferente, que aprovecha la información útil de los objetos y atributos con este tipo de valores, pero penaliza los atributos con valores faltantes en los objetos de cada nodo. Para entender el por qué de la penalización consideremos el siguiente ejemplo. Supongamos que tenemos que elegir entre dos formas de dividir un nodo, en un problema con dos clases, A y B:

- División por atributo 1. (50A, 5B), (2A, 21B) y 100 con el valor faltante.
- División por atributo 2. (70A, 20B), (12A, 74B) y 2 con el valor faltante.

Si ignoramos los valores faltantes y aplicamos una medida de impureza, seleccionaríamos la división 1 como la mejor, pero entonces ninguno de los patrones que obtengamos soportará a los 100 objetos con datos faltantes, lo que puede generar que los

objetos con valores faltantes no puedan ser clasificados por ninguno de los patrones extraídos.

La solución que implementamos fue considerar en la función de evaluación los valores faltantes como otro nodo con impureza máxima. Nótese que esto no significa que se genera un nuevo nodo y que de él se extraen patrones, lo que sería incorrecto, pues se consideraría al valor faltante como un valor más.

En el ejemplo anterior, para el cálculo de la ganancia de información, se consideraría como si existieran tres nodos hijos, y el último tendría 100 objetos con entropía máxima, lo que es igual que si fueran 50 de cada clase.

Los resultados experimentales utilizando esta técnica fueron muy superiores a la variante de ignorar los valores faltantes a la hora de evaluar un nodo, aunque aún no hemos realizamos comparaciones con métodos de estimación de los valores faltantes, lo que se hará en el transcurso de la investigación

5.3. Comparación experimental

En nuestras experimentaciones utilizamos un clasificador basado en la suma de votos por clases, y la clase con mayor suma es la asignada al objeto a clasificar. Estos votos son calculados por el método de estimar la calidad de un EP expuesto previamente. El clasificador obtenido fue comparado contra un conjunto de clasificadores del estado del arte, entre los que se encuentran tanto clasificadores individuales como combinaciones de clasificadores. Los clasificadores escogidos fueron C4.5 [4], kNN ($k = 3$ y $k = 7$), RandomForrest [30] y AdaBoost [29] y se utilizaron las implementaciones que aparecen en Weka [40]. Se escogieron estos clasificadores por ser los que aparecen mas frecuentemente en los artículos publicados sobre patrones emergentes. En todos los casos se realizó validación cruzada con 10 pliegues, excepto en las bases de datos que aparecen divididas en entrenamiento y prueba en el repositorio (los MonksProblem). En todas las pruebas los clasificadores utilizaron exactamente los mismos conjuntos de entrenamiento y prueba.

Las pruebas se realizaron sobre las bases de datos que aparecen en la Tabla 2, tomadas del repositorio de la UCI [27].

Los resultados de la eficacia de cada clasificador en las bases de datos utilizados puede observarse en la Tabla 3. Las primeras columnas representan la eficacia de los respectivos clasificadores, medida como el porcentaje de objetos correctamente clasificados. En el caso de nuestro clasificador las abstenciones se contaron como errores, apareciendo el porcentaje de abstención en la columna “Abs”. El mayor valor de eficacia para cada base de datos aparece en negritas.

Tabla 2. Descripción de las bases de datos utilizadas en los experimentos

Nombre	**# Objetos**	**Distrib. Clases (%)**	**# Atributos**		**Datos faltantes**
			Numéricos	No numéric.	
CreditScreening	690	45%/55%	6	9	5%
BreastCancerWisc	699	65%/35%	-	9	16 objs
CMC	1473	43/23/34	2	7	-
CylinderBand	512	61/29	20	20	302 objs
Hepatitis	155	79/21	6	13	6%
Iris	150	33/33/33	4	-	-
MonksProblem1	432		-	6	-
MonksProblem2	432		-	6	-
MonksProblem3	432		-	6	-
WDBC	569	63/37	30	-	-
WPBC	198	76/24	32	-	4 objs

En la última columna aparece el valor “si” en el caso que existan diferencias significativas entre el resultado de nuestro método y el de mejores resultados del resto, y “no” en caso contrario. Las pruebas de significación estadística fueron realizadas con la prueba T de Student por pares con un nivel de significancia de 0.05 [41]. No se utilizó esta prueba con las bases de datos donde no se realizó validación cruzada (no poseen valor en esta columna).

Como puede observarse en los resultados de la Tabla 3 nuestro método superó al resto en 6 de las 11 bases de datos, en varias de manera significativa.

Una comparación con los clasificadores basados en patrones con algunas bases de datos aparece en la Tabla 4. La segunda columna refleja los resultados del clasificador DeEP [15], que extrae los patrones emergentes al momento de clasificar, mientras que la tercera columna refleja el del clasificador utilizando SJEP [12]. Estos resultados no fueron encontrados ejecutando los algoritmos, sino tomando los que aparecen en los respectivos trabajos, por lo que deben ser interpretados con reserva. No obstante nos parece útil para tener una idea del desempeño de nuestro algoritmo, en comparación con otros basados en patrones.

Tabla 3. Resultados experimentales

Base de Datos	% de Eficacia (correctos/total * 100)						Abst	Signif
	3NN	7NN	Boost	Rand Forrest	C4.5	Nuestro		
CreditScreening	84.15	86.24	**86.28**	85.02	85.08	86.02	0.71	no
BreastCancerWisc	96.53	95.71	95.58	95.93	96.47	**97.63**	0	no
CMC	47.06	48.19	42.52	50.95	50.90	**55.08**	5.14	**si**
CylinderBand	70.16	71.19	72.86	72.19	78.51	**81.73**	0	**si**
Hepatitis	**86.04**	85.08	83.58	81.79	82.37	82.67	5.38	
Iris	96.59	97.21	**97.75**	95.83	95.20	97.67	0	no
MonksProblem1	81.02	76.85	75.00	75.69	88.89	**92.59**	0	-
MonksProblem2	61.34	65.28	60.65	65.05	69.88	**73.61**	0.23	-
MonksProblem3	88.19	92.82	97.22	97.22	96.30	**97.45**	0.23	-
WDBC	**96.27**	95.94	92.32	91.43	94.02	93.46	0	
WPBC	72.30	**75.95**	71.02	74.69	75.51	73.69	0	

Como se observa en la Tabla 4, nuestro método obtiene resultados comparables con el clasificador basado en SJEP, y ambos son superiores a los de DeEP. Este resultado nos parece muy prometedor, pues nos comparamos con las mejores variantes conocidas hasta el momento de ambos clasificadores.

Tabla 4. Resultados de la comparación experimental del nuevo método con otros basados en patrones

Base de Datos	DeEP	Clasif. con SJEP	Nuestro
BreastCancer	96.42	96.96	**97.63**
CreditScreening	84.18	**87.65**	86.02
Hepatitis	81.18	**83.33**	82.67
Iris	96.00		**97.67**

Algunos ejemplos de patrones encontrados por nuestro algoritmo en la base de datos CreditScreening, que contiene información sobre el otorgamiento de créditos a individuos por bancos japoneses, son los siguientes:

```
(Feat 10>1.000) and (Feat 5='cc') (Clase '+':13)
(Feat 10>1.000) and (Feat 5='e') and (Feat 7>0.165) (Clase '+':6)
(Feat 10>1.000) and (Feat 5='ff') and (Feat 1<=39.580) (Clase '-':9)
```

Estos patrones describen dos propiedades que permiten distinguir la clase positiva (con 13 y 6 de ejemplos de soporte) y una propiedad para la clase negativa (con 9 ejemplos). Como puede observarse los patrones emergentes son entendibles por el usuario.

6. Conclusiones

Los clasificadores basados en patrones permiten clasificar objetos con calidad competitiva y a veces superior a otros clasificadores, obteniéndose con ellos modelos

fácilmente legibles e interpretables por los usuarios. No obstante su eficacia está estrechamente relacionada con la calidad de los patrones que utilizan.

El proceso de extracción automática de patrones ha estado desde sus inicios limitado por el costo computacional de las búsquedas exhaustivas. Para ello se han introducidos numerosas soluciones, pero esto se ha hecho al coste de limitar el lenguaje de representación de los patrones.

Como avances preliminares de la investigación proponemos una extensión al lenguaje de representación de patrones emergentes, que permite expresar propiedades más generales, y un algoritmo para encontrarlos en bases de datos con DMI. También se propone una nueva forma de ponderar los votos emitidos por los patrones al momento de clasificar. Las experimentaciones muestran que un clasificador simple basado en estos patrones supera a los clasificadores contra los que se comparó en varias bases de datos de prueba.

Basados en estos resultados preliminares podemos concluir que nuestros objetivos son alcanzables siguiendo la metodología propuesta

7. Referencias

[1] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1982.

[2] S. Haykin, *Neural Networks: A Comprehensive Foundation*: Prentice Hall PTR, 1998.

[3] B. D. Dasarathy, *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques.* Los Alamitos, California: IEEE Computer Society Press, 1991.

[4] J. R. Quinlan, *C4.5: Programs for Machine Learning*: Morgan Kaufmann Publishers Inc., 1993.

[5] J. C. William, "The epistemology of a rule-based expert system: a framework for explanation," Stanford University 1981.

[6] K. Ramamohanarao, J. Bailey, and H. Fan, " Efficient Mining of Contrast Patterns and Their Applications to Classification," in *International Conference on Intelligent Sensing and Information Processing*, 2005, pp. 39-47.

[7] M. N. Bongard, "Solution to geological problems with support of recognition programs," *Sov. Geologia,* vol. 6, pp. 33-50, 1963.

[8] M. Bongard, *Problem of Recognition*. Moscow: Nauka Publishers, 1967.

[9] L. De la Vega-Doria, J. A. Carrasco Ochoa, and J. Ruiz-Shulcloper, "Fuzzy Kora-Omega Algorithm," in *Sixth European Congress on Intelligent Techniques and Soft Computing*, Aachen, Germany, 1998, pp. 1190-1194.

[10] G. Dong and J. Li, "Efficient mining of emerging patterns: discovering trends and differences," in *Proceedings of the fifth ACM SIGKDD international conference on*

Knowledge discovery and data mining San Diego, California, United States: ACM, 1999.

[11] J. Li, G. Dong, and K. Ramamohanarao, "Instance-Based Classification by Emerging Patterns," in *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*: Springer-Verlag, 2000.

[12] H. Fan and K. Ramamohanarao, "Fast Discovery and the Generalization of Strong Jumping Emerging Patterns for Building Compact and Accurate Classifiers," *IEEE Transactions on Knowledge and Data Engineering,* vol. 18, pp. 721-737, 2006.

[13] Y. Sun, K. C. A. Wong, and Y. Wang, "An Overview of Associative Classifiers," in *International Conference on Data Mining (DMIN'06)*, 2006.

[14] K. Ramamohanarao and H. Fan, "Patterns Based Classifiers," in *World Wide Web*. vol. 10: Kluwer Academic Publishers, 2007, pp. 71-83.

[15] J. Li, G. Dong, K. Ramamohanarao, and L. Wong, "DeEPs: A New Instance-Based Lazy Discovery and Classification System," in *Machine Learning*. vol. 54: Kluwer Academic Publishers, 2004, pp. 99-124.

[16] H. Fan and K. Ramamohanarao, "A Bayesian approach to use emerging patterns for classification," in *Proceedings of the 14th Australasian database conference - Volume 17* Adelaide, Australia: Australian Computer Society, Inc., 2003.

[17] J. F. Martínez-Trinidad and A. Guzmán-Arenas, "The logical combinatorial approach to pattern recognition, an overview through selected works," *Pattern Recognition,* vol. 34, pp. 741-751, 2001.

[18] H. Fan and K. Ramamohanarao, "An Efficient Single-Scan Algorithm for Mining Essential Jumping Emerging Patterns for Classification," in *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*: Springer-Verlag, 2002.

[19] A. Inokuchi, T. Washio, and H. Motoda, "An Apriori-Based Algorithm for Mining Frequent Substructures from Graph Data," in *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*: Springer-Verlag, 2000.

[20] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, 1973.

[21] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifier," *Machine Learning,* vol. 2, pp. 131-163, 1997.

[22] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," in *Fourth International Conference on Knowledge Discovery in Databases and Data Mining*, New York, USA, 1998.

[23] D. Meretakis and B. Wuthrich, "Extending naive bayes classifiers using long itemsets," in *Fifth International Conference on Knowledge Discovery and Data Mining*, San Diego, USA, 1999, pp. 165-174.

[24] J. Li, K. Ramamohanarao, and G. Dong, "Combining the Strength of Pattern Frequency and Distance for Classification," in *Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining*: Springer-Verlag, 2001.

[25] H. Inakoshi, T. Ando, A. Sato, and S. Okamoto, "Discovery of Emerging Patterns from Nearest Neighbors," in *International Conference on Machine Learning and Cybernetics*, Beijing, 2002, pp. 1920-1925.

[26] X. Zhang, G. Dong, and R. Kotagiri, "Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* Boston, Massachusetts, United States: ACM, 2000.

[27] C. J. Merz and P. M. Murphy, "UCI Repository of Machine Learning Databases," University of California at Irvine, Department of Information and Computer Science, Irvine, Technical report 1998.

[28] S. J. Nowland and G. E. Hinton, "Evaluation of Adaptive Mixtures of Competing Experts," in *Advances in Neural Information Processing Systems*, R. P. Lippmann, J. E. Moody, and D. S. Touretzky, Eds., 1991, pp. 774-780.

[29] Y. Freund and R. E. Shapire, "A decision-theoretic generalisation of on-line learning and an application to boosting," *Journal of Computer ans System Sciences,* vol. 55, pp. 119-139, 1997.

[30] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 832-844, 1998.

[31] H. Alhammady and K. Ramamohanarao, "The application of emmerging patterns for improving the quality of rare-class classification," in *8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD2004)*, Sydney, Australia, 2004, pp. 207-211.

[32] H. Alhammady and K. Ramamohanarao, "Using emerging patterns and decision trees in rare-class classification," in *4th IEEE International Conference on Data Mining (ICDM 2004)*, Brighton, UK, 2004, pp. 315-318.

[33] M. V. Joshi, "Learning Classifiers Models for Predicting Rare Phenomena." vol. PhD Minnesota, Usa: University of Minnesota, 2002.

[34] P. Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive," in *International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, USA, 1999.

[35] C. X. Ling and C. Li, "Data Mining for Direct Marketing: Problems and Solutions," in *International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 1998.

[36] A. Appice, M. Ceci, C. Malgieri, and D. Malerba, "Discovering Relational Emerging Patterns," in *AI*IA 2007: Artificial Intelligence and Human-Oriented Computing*, 2007, pp. 206-217.

[37] R. S. Michalski and R. Stepp, "Revealing Conceptual Structure in Data by Inductive Inference," in *Machine Intelligence*. vol. 10, D. Michie, J. E. Hayes, and H. H. Pao, Eds. New York: Ellis Horwood Ltd, 1982, pp. 173-196.

[38] J. Han and J. Pei, "Mining frequent patterns by pattern-growth: methodology and implications." vol. 2: ACM, 2000, pp. 14-20.

[39] R. D. Wilson and T. R. Martinez, "Improved Heterogeneous Distance Functions," *Journal of Artificial Intelligence Research,* vol. 6, pp. 1-34, 1997.

[40] I. Wittn, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunnigham, "Weka: Practical machine learning tools and techniques with Java implementations," in *Emerging Knowledge Engineering and Connectionist-Based Information Systems*, 1999, pp. 192-196.

[41] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms." vol. 10: MIT Press, 1998, pp. 1895-1923.