Método rápido de preprocesamiento para clasificación en conjuntos de datos no balanceados

Liliana Puente-Maury[1], Asdrúbal López-Chau[2], William Cruz-Santos[2], Lourdes López-García[2]

[1] Universidad Autónoma Metropolitana, Unidad Cuajimalpa, Distrito Federal, México

[2] Universidad Autónoma del Estado de México, CU UAEM Zumpango, Estado de México, México

alchau@uaemex.mx

Resumen. El problema de desbalance en clasificación se presenta en conjuntos de datos que tienen una cantidad grande de datos de cierto tipo (clase mayoritaria), mientras que el número de datos del tipo contrario es considerablemente menor (clase minoritaria). En este escenario, prácticamente todos los métodos de clasificación presentan un bajo desempeño. En este artículo se propone un nuevo método de preprocesamiento, que utiliza un enfoque similar a las técnicas de basadas en enlaces Tomek, pero cuyo tiempo de ejecución es dramáticamente reducido con respecto al cálculo por fuerza bruta, comúnmente utilizado en dichas técnicas. Los resultados obtenidos en los experimentos demuestran la efectividad del método propuesto para mejorar las áreas de las curvas ROC y PRC de métodos de clasificación aplicados a conjuntos de datos reales no balanceados.

Palabras clave: Tomek, desbalance, clasificación.

1. Introducción

En clasificación, el problema de desbalance se presenta de manera natural en diversos dominios del mundo real [9]. Por ejemplo, en condiciones normales, la cantidad de operaciones fraudulentas en transacciones bancarias es considerablemente menor a las operaciones no fraudulentas. En este caso, la clase minoritaria correspondería a las primeras transacciones mencionadas, mientras que la clase mayoritaria correspondería a las segundas. En medicina, se ha observado que ciertas enfermedades afectan a un número reducido de personas; por lo que el número de expedientes de personas que las padecen es reducido, comparado con el total de expedientes médicos.

En este tipo de escenarios, es de vital importancia que los sistemas de reconocimiento automático puedan predecir correctamente instancias de clase minoritaria y, al mismo tiempo, que no dañen la precisión de las predicciones para la clase mayoritaria.

El problema de desbalance es complejo, y no solamente depende de la proporción que existe entre el número de instancias de cada clase, dicho problema es conocido como "desbalance entre clases" [1,7]. La complejidad de los datos juega un papel importante en este tipo de problemas. Entiéndase ésta como el traslape entre clases, la falta de datos representativos en algunas regiones del espacio de entrada o la existencia de subconceptos [6]. Cuando dentro de un problema de clasificación existen subconceptos que contienen pocas instancias, se presenta lo que se conoce como el "desbalance al interior de las clases".

Para mejorar el desempeño de sistemas de reconocimiento de patrones en conjuntos de datos desbalanceados, se han propuesto soluciones que intentan balancear o limpiar los datos antes de aplicarlos a métodos de clasificación existentes. Estas soluciones son llamadas métodos externos y trabajan con los datos en una etapa de preprocesamiento. En otras propuestas, se modifican los algoritmos de clasificación con la finalidad de incluir en ellos un mecanismo para hacer que las instancias de la clase minoritaria sean consideradas de mayor importancia que el resto, o en otras palabras, se fuerza a que el método de clasificación realice una generalización que favorezca a la clase minoritaria.

Uno de los primeros métodos externos para reducir el desbalance al interior de las clases fue la utilización de *enlaces Tomek*. De manera informal, un enlace Tomek está conformado por dos instancias de clase contraria (una de clase minoritaria y la otra de la mayoritaria) cuya distancia es la menor con respecto a cualquier otra instancia del conjunto de datos. La idea subyacente de los métodos que usan enlaces Tomek, consiste en identificar y eliminar del conjunto de datos, aquellos objetos de clase mayoritaria que se encuentran ya sea cerca de la frontera de decisión o al interior de un subconcepto perteneciente a la clase minoritaria. Aunque este método funciona en la mayoría de los casos, una desventaja es que su complejidad es cuadrática en espacio de almacenamiento, y casi cúbica en tiempo de cómputo. Otro método ampliamente utilizado es SMOTE [2], que agrega instancias creadas artificialmente al conjunto de datos. Esto hace poco práctico la utilización directa de enlaces Tomek y de SMOTE para datos grandes.

En este artículo, se propone un nuevo método externo para preprocesamiento de conjuntos de datos no balanceados. El método toma una idea similar al que emplea enlaces Tomek, sin embargo, en nuestro caso, la complejidad computacional es considerablemente menor, tal y como se demuestra en los resultados de los experimentos realizados. Nuestro método realiza particiones el espacio de entrada en regiones de baja entropía, y luego detecta regiones adyacentes de clase contraria. Mediante esta técnica, la cantidad de instancias candidatas a formar enlaces Tomek se reduce considerablemente. Para demostrar la efectividad de la propuesta presentada en este artículo, se utilizan seis conjuntos de datos reales disponibles públicamente en Internet, y se emplea al clasificador C4.5 para medir su desempeño antes y después de aplicar nuestros algoritmos. El resto del artículo está organizado como sigue. En la sección 2 se muestran algunas definiciones de las medidas utilizadas para comparar el desempeño de los métodos de clasificación, se describen a los enlaces Tomek y a los árboles de decisión. En la sección 3 se presentan los algoritmos desarrollados que implementan el método

propuesto y se muestra el análisis de complejidad del mismo. Los resultados de los experimentos se presentan en la sección 4. El artículo termina con conclusiones y líneas de investigación futuras.

2. Preliminares

En esta sección se presentan algunas medidas estadísticas relacionadas con la medición del desempeño de clasificadores en conjuntos de datos desbalanceados. También se incluye la definición de los enlaces Tomek y el algoritmo básico para su detección. Además, en la última parte de esta sección, se muestran los árboles de decisión, utilizados en una parte del método propuesto.

2.1. Medidas utilizadas en clasificación de conjuntos de datos no balanceados

Las métricas utilizadas para medir el desempeño de métodos de clasificación sobre conjuntos de datos balanceados, tales como precisión de clasificación y error medio, resultan inadecuadas para el problema de clasificación en datos no balanceados [5]. Por ello, se han desarrollado nuevas métricas para este tipo de escenarios. A continuación, se presentan las más importantes.

Sea P el conjunto de datos de clase positiva (p) (conjunto de instancias de la clase minoritaria), y N el conjunto de datos de clase negativa (n) (conjunto de instancias de clase mayoritaria). Definamos la función $f_C : X \to \{n, p\}$ como:

$$f_C(x) = \begin{cases} p \text{ Si } C \text{ clasifica a } x \in X \text{ como positivo,} \\ n \text{ si } C \text{ clasifica a } x \in X \text{ como negativo} \end{cases}$$

donde C, es el método de clasificación utilizado, y X es el conjunto de instancias a clasificar. Entonces, los conjuntos de los falsos positivos y el de los falsos negativos se pueden formalizar como sigue:

$$FP = \{x \in X \ | x \in N, f_C(x) = p\} \text{ y } FN = \{x \in X \ | x \in P, f_C(x) = n\}$$

De manera análoga, el conjunto de los verdaderos positivos y el de los verdaderos negativos se define como:

$$TP = \{x \in X \ | x \in P, f_C(x) = p\} \text{ y } TN = \{x \in X \ | x \in N, f_C(x) = n\}$$

Dos medidas importantes son la tasa de verdaderos positivos (TP_{rate}) y la de falsos positivos (FP_{rate}), calculadas como:

$$TP_{rate} = {}^{TP}/_{(TP+FN)}$$

$$FP_{rate} = {}^{FP}/_{(FP+TN)}$$

La precisión se obtiene usando la ecuación (1), y se puede interpretar como la cantidad de objetos que fueron etiquetados como positivos(negativos), son realmente de ese tipo.

$$Precision = {}^{TP}/_{(TP+FP)}. \tag{1}$$

El recuerdo (*Recall*) es una medida que ofrece un panorama acerca de la relación entre la cantidad de muestras con etiqueta positiva/negativa que fueron predichos como positivos/negativos, y se obtiene con la ecuación (2)

$$Recall = {}^{TP}/_{(TP+FN)} \tag{2}$$

F-Measure (ecuación (3)) combina tanto la precisión como el recuerdo.

$$F - Measure = {}^{(1+\beta)^2 \cdot Recall \cdot Precision}/_{(\beta^2 \cdot Recall + Precision)}, \tag{3}$$

donde β es un coeficiente para ajustar la importancia relativa de la precisión contra el *Recall* (usualmente $\beta = 1$).

La curva *ROC* (Receiver Operating Curve) se forma graficando TP_{rate} contra FP_{rate}, por lo que un punto en este espacio corresponde al desempeño de un algoritmo de clasificación en una distribución dada. Esta curva es muy útil, pues ofrece una representación visual del compromiso (*trade-off*) entre los beneficios (reflejados por los TP) y los costos (reflejados por los FP) de la clasificación [6].

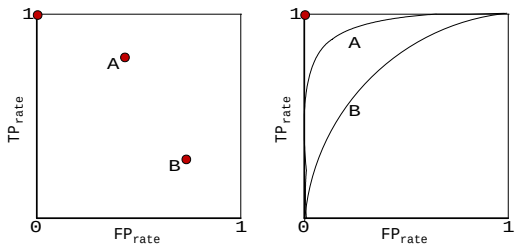

Fig. 1: Espacio ROC para clasificadores *hard-type* (izquierda) y *soft-type* (derecha).

Si el clasificador es *hard-type* (aquellos en la que la predicción indica solamente la clase a la que pertenece una instancia, no su grado de pertenencia a dicha clase), entonces producirá un par (TP_{rate}, FP_{rate}) que corresponde a un punto en el espacio *ROC*. El par $(0, 1)$ corresponde a una clasificación perfecta (costo cero, beneficio máximo), así que un clasificador será mejor que otro cuanto más cerca se encuentre del punto $(0, 1)$. En la Figura 1 (izquierda), el clasificador asociado al punto A es mejor que aquél asociado a B. Si el clasificador es continuo o *soft-type* (o sea, que produce un valor numérico continuo para representar la confianza de una instancia que pertenece a la clase predicha), se puede utilizar un

umbral para producir una serie de puntos en el espacio ROC. Esta técnica puede generar una curva en lugar de un solo punto ROC. En este caso, el clasificador que tenga una mayor área bajo su curva, será mejor. En la Figura 1 (derecha), el clasificador asociado a la curva A es mejor que aquél asociado a B.

El coeficiente de correlación de Matthews, ecuación (4), se usa como una medida de la calidad de las clasificaciones de dos clases. Devuelve un valor entre -1 (ninguna relación entre predicción y observación) y 1 (predicción perfecta).

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \tag{4}$$

Las curvas PRC (Precision-Recall Curve) pueden ofrecer mayor información sobre la valoración del desempeño en el caso de conjuntos de datos altamente sesgados; por esto muchos trabajos actuales usan este tipo de curvas para evaluaciones de desempeño y comparaciones. Estas curvas se definen graficando la tasa de precisión contra la tasa de *Recall*. Las curvas PR tienen una estrecha relación con las curvas ROC: una curva domina en el espacio ROC si, y sólo si, domina en el espacio PR[6][3].

2.2. Enlaces Tomek

Los enlaces Tomek se definen usando el concepto de distancia. Típicamente se utiliza la Euclidiana; sin embargo, es posible utilizar cualquier otra función d que satisfaga las condiciones de métrica sobre un conjunto P, $d : P \times P \mapsto R$,

1. $d(a, b) \geq 0 \ \forall a, b \in P$.
2. $d(a, b) = d(b, a) \ \forall a, b \in P$.
3. $d(a, b) \leq d(a, c) + d(c, b) \ \forall a, b, c \in P$.
4. $d(a, b) = 0$ implica que $a = b$.

Dado un conjunto de datos X con atributos numéricos, un par de objetos α, β, forman un enlace Tomek si satisfacen la siguiente condición:

$$d(\alpha, \beta) < d(\alpha, \gamma) \text{ y } d(\alpha, \beta) < d(\beta, \gamma), \ \forall \gamma \in X, \tag{5}$$

donde $d(a, b)$ es la distancia entre los objetos a y b, $\gamma \in X$.

Si los objetos α y β forman un enlace Tomek, entonces uno de ellos puede ser considerado como ruido, o ambos se encuentran cerca de la frontera de decisión. La Figura 2 ejemplifica esta idea.

Para conjuntos de datos desbalanceados, a cada objeto x de clase minoritaria se le calcula el objeto y de clase mayoritaria que forme un enlace Tomek con él, y se elimina y del conjunto de datos. El Algoritmo 1 muestra el pseudocódigo que implementa esta idea. En los algoritmos mostrados, X es el conjunto de datos, con $X = X^+ \cup X^-$ y $X^+ \cap X^- = \emptyset$, donde X^+ y X^- son el conjunto de instancias de clase minoritaria y mayoritaria, respectivamente.

[3] Una curva PRC dominante, se encuentra en la parte superior derecha del espacio PR.

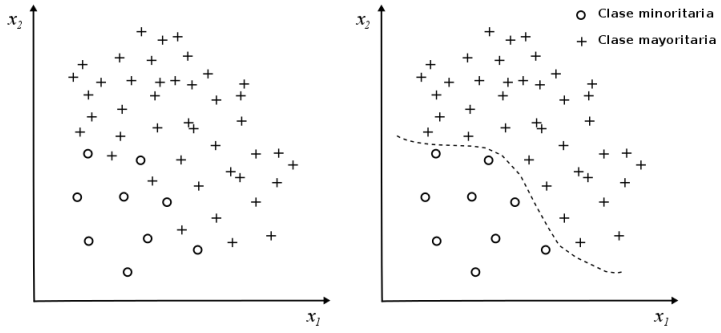

Fig. 2: Aplicación de enlaces de Tomek para eliminar instancias de clase mayoritaria.

Algorithm 1: Preprocesamiento usando enlaces Tomek.

Input : X: Conjunto de datos
Output: X_{proc}: Conjunto de datos procesado

begin
  $X_{proc} \leftarrow X$`//Copia de` X;
  $T_{Links} \leftarrow$ Pares de objetos que forman enlaces Tomek (Algoritmo 2);
  foreach $x_i \in T_{links}$ && $x_i \in X^-$ **do**
    $X_{proc} \leftarrow X_{proc} - \{x_i\}$`//Elimina` x_i
  end
  return X_{proc}
end

El Algoritmo 2 muestra el pseudocódigo para determinar todos los enlaces Tomek de un conjunto de datos. La implementación simple usando fuerza bruta tiene un orden de complejidad aparente de $O(n^2)$ (doble ciclo: el exterior de 1 a n y el interior que recorre los restantes $n-1$ puntos). Sin embargo, para determinar que se cumpla la condición (5), se requiere un ciclo adicional. Por lo que el peor caso es $O(n^3)$. Esta es una de las principales limitaciones prácticas al aplicar enlaces de Tomek en conjuntos de datos grandes.

2.3. Árboles de decisión

El método propuesto utiliza un árbol de decisión para realizar particiones en el espacio de entrada. En esta subsección, se realiza una breve revisión de este clasificador. Los árboles de decisión son un método de aprendizaje supervisado cuya estructura puede ser representada por un grafo acíclico que forma un árbol [4]. Las partes principales de dicha estructura son nodos (raíz, internos y hojas) y vértices que unen a los nodos.

Algorithm 2: Cálculo de objetos que forman enlaces Tomek.

Input :
X: Conjunto de datos
Output:
T_{links}: Colección de pares de objetos que forman enlaces Tomek

```
begin
    X+ = {x_i ∈ X tal que y_i = +1}//Clase minoritaria;
    X− = {x_i ∈ X tal que y_i = −1}//Clase mayoritaria;
    foreach x_i ∈ X− do
        foreach x_j ∈ X+ do
            dT ← d(x_i, xi);
            if ¬∃δ ∈ X tal que dT > d(δ, x_j) o d(δ, xj) then
                T_links ← T_links ∪ (x_i, x_j);
            end
        end
    end
    return T_links
end
```

Se ha demostrado que crear un árbol de decisión óptimo en términos de tamaño y desempeño es un problema NP completo [3,8], es por ello que se utilizan heurísticas que permiten construir árboles de una manera más eficiente. El método básico para *inducir* un árbol de decisión a partir de datos, consiste en seleccionar un atributo en el cual se divida en dos un conjunto de datos, de tal forma que después de separado, cada parte sea "más pura" que el conjunto antes de la partición. Elegir el mejor atributo en el cual realizar la división, implica medir la pureza de un conjunto de instancias. Las medidas más populares utilizadas en árboles de decisión son la entropía, ganancia de información, la tasa de ganancia y el índice Gini. Las ecuaciones (6), (7), (8) y (9) muestran cómo calcular estas medidas.

$$Entropy(X) = -\sum_{i=1}^{n} p_i log_2(p_i) \tag{6}$$

$$Gain(X, A) = Entropy(S) - \sum_{v \in valores(A)} p_i log_2(p_i) \tag{7}$$

$$GainRatio(A) = \frac{Gain(A)}{-\sum_{j=1}^{v} \frac{\|D_j\|}{\|D\|} log_2\left(\|D_j\| \|D\|\right)} \tag{8}$$

$$Gini(X) = 1 - \sum_{i=1}^{m} p_i^2 \tag{9}$$

3. Método propuesto

La implementación directa del cálculo de enlaces Tomek (Algoritmo 2) tiene una complejidad computacional superior a la cuadrática. Para evitar este elevado costo, se desarrolló un nuevo método que realiza un preprocesado rápido a los datos similar al que emplea enlaces Tomek, y que permite mejorar el desempeño de algoritmos de clasificación.

La idea del método propuesto en este artículo es simple de implementar, pero efectiva y eficiente. El procedimiento general puede resumirse en los siguientes pasos:

1. Particionar el espacio de entrada en regiones de baja entropía,
2. Numerar las particiones,
3. Determinar las particiones adyacentes de cada partición encontrada,
4. Detectar las instancias más cercanas que pertenecen a dos particiones adyacentes de clase contraria,
5. Repetir el paso 4 para cada partición,
6. Eliminar del conjunto de datos aquellas instancias detectadas en el paso 4 que sean de clase mayoritaria.

Para el particionado del espacio de entrada en regiones de baja entropía, el método presentado utiliza un árbol de decisión C4.5. La idea es realizar esta tarea de una forma rápida, de tal manera que cada región dividida del espacio de entrada contenga la mayor cantidad de instancias de una clase, y al mismo tiempo, la menor cantidad de instancias de clase contraria.

Dos observaciones importantes, que permitieron diseñar un nuevo método para detectar rápidamente instancias de clase mayoritaria que pueden ser eliminadas del conjunto de datos, son las siguientes:

1. En una región del espacio de entrada con entropía mínima, los enlaces Tomek se presentan cerca de regiones adyacentes con instancias de clase contraria.
2. Dos regiones adyacentes con entropía mínima y de la misma clase, no pueden formar enlaces Tomek entre ellas.

Una parte esencial consiste en determinar cuáles particiones del espacio de entrada son adyacentes. Para ello, se diseñaron dos algoritmos. El primero de ellos, Algoritmo 3, realiza un recorrido por un árbol de decisión recopilando información sobre los límites que definen cada hoja del árbol. Las hojas corresponden a las regiones de baja entropía. El segundo, Algoritmo 4, detecta todas las particiones adyacentes de cada partición, haciendo una búsqueda en los límites determinados durante el recorrido del árbol. La Figura 3 muestra un ejemplo hipotético de un espacio de entrada particionado, y la estructura del árbol que representa las particiones. La Tabla 1 muestra los límites de cada partición encontrados por los Algoritmos 3 y 4. Las columnas $B_{i,L}$ y $B_{i,H}$ corresponden a los límites inferior (L) y superior (H) del atributo i-ésimo, respectivamente.

El análisis de complejidad de nuestro método es el siguiente. Sin perder generalidad, se puede suponer que los enlaces Tomek se determinarán desde

Algorithm 3: Cálculo de los límites de las hiper cajas.

Input : $\mathcal{T}$: Un árbol de decisión inducido a partir de X
Output: M: Matrix con los límites de cada hoja de $\mathcal{T}$

begin
  Crear vector $B \in R^{2d}$;
  Inicializar con $-\infty$ los elementos de B con índice 1 a d;
  Inicializar con $+\infty$ los elementos de B con índice d+1 a 2d;
  Invocar a DescubreLímites(B, M) desde nodo raíz;
  regresar M;
end

DescubreLímites(B,M**)**;
if *nodo visitado es hoja* **then**
  Agregar B como la última fila de M;
end
else
  Crear B_L y copiar límites desde B;
  B_L: Cambiar h_{ij} usando el índice de atributo y el valor de particionado del nodo actual;
  Invocar a DescubreLímites(B_L, M) con el hijo izq. del nodo actual;
  Crear B_R y copiar límites desde B;
  B_R: Cambiar l_{ij} usando el índice de atributo y el valor de particionado del nodo actual;
  Invocar a DescubreLímites(B_R, M) con el hijo derecho del nodo actual;
end

return

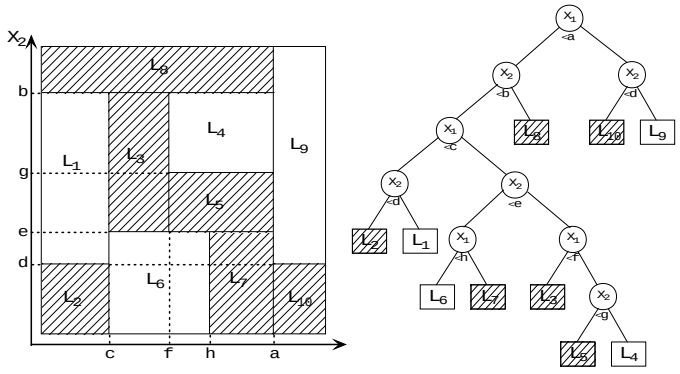

Fig. 3: Particiones en el espacio de entrada.

el centro de una partición hacia el centro de otra. Entonces, habría que calcular las distancias desde dicho centro, hacia cada centro de las particiones adyacentes que contienen instancias de clase contraria. Para conjuntos de datos balanceados, se puede suponer que las instancias están distribuidas de manera uniforme

dentro de las particiones, y que estas tienen una entropía mínima. Bajo estas suposiciones, el número de instancias en cada partición es aproximadamente n/m, donde m es la cantidad total de particiones. El peor caso ocurre cuando todas las particiones son adyacentes entre sí, y la mitad de ellas contienen instancias de clase mayoritaria y la otra mitad de clase minoritaria. El número de distancias que se tendrían que calcular sería el especificado por la ecuación (10).

Algorithm 4: Determinación de los vecinos de una hiper caja.

Input : M: Matriz de límites (Algoritmo 3)
L: Hiper caja referencia
Output: L_{vecino}: Lista de vecinos de L

```
begin
    for i=1 to d do
        Crear lista L_vecino con todas las cajas que compartan un límite con la
        caja L (usar M para detectar esto);
        foreach elemento e_j de L_1 do
            if e_j forma espacio conectado con caja L then
                Agregar e_j a L_vecino
            end
        end
        Eliminar elementos repetidos en L;
        regresar L_vecino;
    end
end
```

Tabla 1: Limites de las particiones.

Partición	$B_{1,L}$	$B_{1,H}$	$B_{2,L}$	$B_{2,H}$
L_1	$-\infty$	c	d	b
L_2	$-\infty$	c	$-\infty$	d
L_3	c	f	e	b
L_4	f	a	g	b
L_5	f	a	g	b
L_6	c	h	$-\infty$	e
L_7	h	a	$-\infty$	e
L_8	$-\infty$	a	b	∞
L_9	a	∞	d	∞
L_{10}	a	∞	$-\infty$	d

$$\left(\frac{m}{2}\right)\left(\frac{m}{2}\right)\left(\frac{n}{m}\right)\left(\frac{n}{m}\right) = \left(\frac{n^2}{4}\right) \tag{10}$$

Lo que da una complejidad de $O(n^2)$. En estas condiciones, el método propuesto tiene una complejidad computacional similar al caso de cálculo por fuerza bruta. Sin embargo, en la práctica hemos observado que las cosas ocurren rara vez como en el peor caso. En general, lo que sucede es lo siguiente:

1. El número de particiones cambia de un conjunto de datos a otro. Atribuimos esto a la diferencia entre complejidad de los conceptos en ellos [6].
2. En conjuntos de datos no balanceados, la cantidad de particiones que contienen instancias de clase minoritaria es significativamente menor a las particiones con instancias de clase contraria.
3. Algunas particiones contienen muchas instancias, mientras que otras contienen pocas de ellas.
4. No todas las particiones son vecinas entre sí. Más aún, la mayoría de las particiones de clase mayoritaria tienen particiones adyacentes con clase similar.

Como el número de particiones de clase minoritaria es pequeño, la cantidad de distancias que se calcula es mucho menor al calculado por el método de fuerza bruta. Esto hace que nuestra propuesta sea superior en velocidad.

4. Resultados

Para probar el desempeño del método propuesto, y realizar comparaciones con el método de fuerza bruta, se utilizaron los conjuntos de datos mostrados en la Tabla 3. Estos conjuntos de datos se encuentran disponibles públicamente en el repositorio Keel $http://sci2s.ugr.es/keel/datasets.php$.

Tanto el método propuesto como el de fuerza bruta fueron implementados en lenguaje de programación Java. Se utilizó Weka como herramienta para cargar archivos, realizar evaluaciones y el algoritmo C4.5 (llamado J48 en Weka) para particionado del espacio de entrada. Todos los experimentos fueron ejecutados en una computadora con las siguientes características: Procesador Intel i7 3720QM 2.60 GHz, 8.0 GB en RAM, Sistema Operativo Windows 7 de 64 bits.

Para validar los resultados, se repitieron 100 veces los experimentos en cada conjunto de datos utilizado. En cada ejecución se eligieron aleatoriamente el 70 % de los datos para entrenamiento, y el 30 % restante se usó para prueba. Los resultados presentados en la Tabla 2 corresponden al promedio de ese número de ejecuciones. Una vez procesados los datos, el método C4.5 fue aplicado al conjunto de datos con las instancias de clase mayoritaria eliminadas. Por razones de espacio, sólo se presentan los resultados con este clasificador. El significado de las columnas de la Tabla 2 es el siguiente: La columna T representa el tiempo de preprocesamiento en mili segundos; TP es la tasa de verdaderos positivos; FP es la tasa de falsos positivos; Prec representa la precisión; Recall es el recuerdo con respecto a la clase indicada en la última columna; F-M es F-Measure; MCC es el coeficiente phi; ROC es el área bajo la curva ROC y PRC es el área bajo la curva PR. Las filas *Ninguno*, son los resultados sin preprocesamiento; las filas *Propuesta* corresponden a los resultados obtenidos con la aplicación del método

Tabla 2: Resultados obtenidos en los experimentos.

Preprocesamiento Método	T(ms)	TP	FP	Prec	Recall	F-M	MCC	ROC	PRC	Class
Conjunto de datos **car-good**										
Ninguno	NA	0.954	0.033	0.557	0.954	0.698	0.713	0.962	0.589	Positiva
Ninguno		0.967	0.046	0.998	0.967	0.982	0.713	0.962	0.997	Negativa
Propuesta	**62.134**	0.969	0.038	0.518	0.969	0.671	0.691	0.965	0.526	Negativa
Propuesta		0.962	0.031	0.999	0.962	0.980	0.691	0.965	0.998	Positiva
Tomek	18,303.949	1.000	1.000	0.040	1.000	0.076	0.000	0.500	0.040	Negativa
Tomek		0.000	0.000	0.000	0.000	0.000	0.000	0.500	0.960	Positiva
Conjunto de datos **dermatology-6**										
Ninguno	NA	0.965	0.400	0.398	0.965	0.468	0.446	0.782	0.377	Positiva
Ninguno		0.600	0.035	0.648	0.600	0.621	0.446	0.782	0.977	Negativa
Propuesta	**1.957**	0.958	0.283	0.457	0.958	0.536	0.529	0.837	0.432	Positiva
Propuesta	-	0.717	0.042	0.777	0.717	0.744	0.529	0.837	0.982	Negativa
Tomek	135.002	0.966	0.410	0.392	0.966	0.462	0.438	0.778	0.372	Positiva
Tomek		0.590	0.034	0.638	0.590	0.612	0.438	0.778	0.976	Negativa
Conjunto de datos **ecoli-0-1-4-7_vs_5-6**										
Ninguno	NA	0.913	0.746	0.120	0.913	0.198	0.101	0.584	0.112	Positiva
Ninguno		0.254	0.087	0.351	0.254	0.291	0.101	0.584	0.938	Negativa
Propuesta	**1.468**	0.897	0.661	0.138	0.897	0.222	0.144	0.619	0.129	Positiva
Propuesta		0.339	0.103	0.489	0.339	0.393	0.144	0.619	0.943	Negativa
Tomek	14.531	0.895	0.649	0.144	0.895	0.230	0.151	0.624	0.132	Positiva
Tomek		0.351	0.105	0.490	0.351	0.400	0.151	0.624	0.943	Negativa
Conjunto de datos **glass-1**										
Ninguno	NA	0.981	0.914	0.372	0.981	0.537	0.087	0.535	0.375	Positiva
Ninguno		0.086	0.019	0.335	0.086	0.135	0.087	0.535	0.670	Negativa
Propuesta	**1.951**	0.978	0.902	0.375	0.978	0.539	0.098	0.540	0.379	Positiva
Propuesta		0.098	0.022	0.367	0.098	0.153	0.098	0.540	0.673	Negativa
Tomek	33.132	0.980	0.913	0.372	0.980	0.536	0.086	0.537	0.378	Positiva
Tomek		0.087	0.020	0.340	0.087	0.136	0.086	0.537	0.671	Negativa
Conjunto de datos **vowel0**										
Ninguno	NA	0.958	0.382	0.337	0.958	0.450	0.419	0.791	0.351	Positiva
Ninguno		0.618	0.042	0.825	0.618	0.694	0.419	0.791	0.962	Negativa
Propuesta	**4.804**	0.957	0.406	0.343	0.957	0.450	0.413	0.781	0.355	Positiva
Propuesta		0.594	0.043	0.795	0.594	0.665	0.413	0.781	0.961	Negativa
Tomek	274.320	0.958	0.376	0.338	0.958	0.452	0.422	0.794	0.352	Positiva
Tomek		0.624	0.042	0.835	0.624	0.702	0.422	0.794	0.963	Negativa

presentado en este artículo; las filas *Tomek* corresponden a las aplicadas con el Algoritmo 1 con el enfoque de fuerza bruta.

Como puede observarse, el método de preprocesamiento propuesto es eficiente, y permite mejorar el desempeño del algoritmo de clasificación C4.5 para conjuntos de datos no balanceados. Esto coincide con los resultados encontrados en [10], donde se indica que el preprocesamiento contribuye a que C4.5 obten-

Tabla 3: Conjuntos de datos utilizados en los experimentos.

Nombre	Tamaño	Atributos	Clase mayoritaria	Clase minoritaria	RI
car-good_vs_3-6-8	1,728	6	905	99	9.14
cleveland-0_vs_4	173	13	160	13	12.31
dermatology-6	358	34	338	20	16.90
ecoli-0-1-4-7_vs_5-6	332	6	307	25	12.28
glass1	214	10	138	76	1.82
vowel0	988	13	898	90	9.98

ga mejores resultados. Aunque no se presentan resultados, otros clasificadores también son beneficiados con este tipo de procesamiento a los datos.

5. Conclusiones

El problema de clasificación en conjuntos de datos no balanceados representa actualmente un reto importante para las comunidades científicas de inteligencia artificial, minería de datos y aprendizaje automático. Son varios los factores que hacen que un problema de este tipo sea complicado, por ejemplo, desbalance entre las clases, desbalance al interior de las clases e instancias anómalas. Los métodos externos realizan un preprocesamiento a los conjuntos de datos no balanceados para cumplir con uno o más de los siguientes objetivos: balancear el conjunto de datos, quitar instancias consideradas como ruido, eliminar traslape entre clases o buscar prototipos que representen el conjunto de datos de una manera que sea fácil de procesar por métodos de clasificación o agrupamiento.

En este artículo, se presenta un nuevo método rápido de preprocesamiento para conjuntos de datos no balanceados. El método sigue una idea similar a los que usan enlaces Tomek, sin embargo, el tiempo de ejecución es dramáticamente reducido. Los resultados obtenidos con el método de clasificación empleado (C4.5) muestran que, una vez preprocesado el conjunto de datos, el desempeño de C4.5 es mejorado. Como trabajo futuro, se plantea la aplicación el método en conjuntos de datos gigantes (Big Data) y la modificación para flujos de datos de alta velocidad.

Referencias

1. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. SIGKDD Explor. Newsl. 6(1), 20–29 (Jun 2004)
2. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: Smote: Synthetic minority oversampling technique. Journal of Artificial Intelligence Research 16, 321–357 (2002)
3. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York, NY, USA (1979)

4. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edn. (2011)
5. He, H.: Self-Adaptive Systems for Machine Intelligence. Wiley (2011)
6. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. on Knowl. and Data Eng. 21(9), 1263–1284 (Sep 2009)
7. Hulse, J.V., Khoshgoftaar, T.: Knowledge discovery from imbalanced and noisy data. Data & Knowledge Engineering 68(12), 1513–1542 (2009)
8. Hyafil, L., Rivest, R.L.: Constructing optimal binary decision trees is np-complete. Inf. Process. Lett. 5(1), 15–17 (1976)
9. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study. Intelligent Data Analysis 6(5), 429 (2002)
10. Luengo, J., Fernandez, A., Herrera, F., Herrera, F.: Addressing data-complexity for imbalanced data-sets: A preliminary study on the use of preprocessing for c4.5. In: Intelligent Systems Design and Applications, 2009. ISDA '09. Ninth International Conference on. pp. 523–528 (2009)