Método de clasificación multiclase utilizando Dicotomías

Por

Miriam Mónica Duarte Villaseñor

Tesis sometida como requisito parcial para obtener
el grado de

MAESTRA EN CIENCIAS EN LA ESPECIALIDAD DE CIENCIAS COMPUTACIONALES

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Tonantzintla, Puebla
Febrero 2012

Supervisada por:

Dr. Jesús Ariel Carrasco Ochoa
Dr. José Francisco Martínez Trinidad

Dedico este trabajo

A mi mamá Magdalena por sus enseñanzas,
educación y apoyo que siempre me acompaña.

A mi tío Beto por el apoyo y cariño
que toda la vida me da.

Agradecimientos

Expreso mi más sincero agradecimiento a mis asesores de tesis Dr. Jesús Ariel Carrasco Ochoa y Dr. José Francisco Martínez Trinidad, por su infinita paciencia, apoyo y orientación durante el desarrollo de este trabajo de tesis, gracias por todos sus consejos y por confiar en mi.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACyT) por el financiamiento otorgado para el desarrollo de esta tesis.

A mi familia, que siempre me ha apoyado, y me anima a seguir adelante.

A todos mis compañeros (docegen), que estuvieron compartiendo esta grata experiencia y en especial a Alejandro T., Alejandro R., Jaime y Adrian.

A mi amigo MCP, por haberme acompañado y apoyado cuando más lo necesite.

Resumen

La clasificación supervisada es una de las tareas abordadas en el reconocimiento de patrones, con aplicación en diferentes áreas como medicina, astronomía, economía, entre otras. En estas áreas, comúnmente aparecen problemas multiclase, es decir, problemas con más de dos clases; en los cuales cada objeto es descrito por atributos numéricos y no numéricos (datos mezclados).
En este trabajo de tesis se propone un nuevo método de clasificación multiclase utilizando dicotomías anidadas, el cual resuelve problemas multiclase por medio de clasificadores binarios (de dos clases). Las dicotomías anidadas forman árboles binarios en los cuales, en cada nodo del árbol, se dividen las clases en dos grupos y se construye un clasificador binario para separar estos grupos. Este proceso se repite recursivamente hasta tener una sola clase en cada hoja del árbol.
La mayoría de los métodos para construir dicotomías anidadas lo hacen de forma aleatoria, lo cual no garantiza encontrar una buena dicotomía. Es por eso que, en esta tesis, se proponen nuevos métodos no aleatorios para construir dicotomías anidadas, siguiendo la idea de separar en los niveles superiores las clases más fáciles de separar y en los niveles inferiores las clases más difíciles, con el objetivo de reducir los errores en los niveles superiores; ya que si se comete un error en un nivel, éste no podrá ser corregido en los niveles posteriores y el objeto quedará mal clasificado.
Para evaluar el desempeño de los métodos propuestos para construir dicotomías anidadas, se compararon los resultados de precisión en diferentes bases de datos del repositorio UCI, contra los resultados obtenidos con métodos que construyen las dicotomías anidadas de forma aleatoria.

Abstract

Supervised classification is one of the main issues in pattern recognition, with applications in different fields such as medicine, astronomy, and economy, among others. In these fields, it is common to deal with multiclass problems, i.e., problems involving more than two classes; where objects are described through quantitative and qualitative attributes (mixed data).
In this thesis, we propose a new method for multiclass classification using nested dichotomies, which solve multiclass problems using binary classifiers (for two class problems). A nested dichotomy consists in a binary tree, where each node divides its classes in two groups, and builds a binary classifier for separating these groups. This process is recursively repeated until each leaf contains only one class.
Most of the methods for building nested dichotomies use a random strategy, which does not guarantee finding a good nested dichotomy. For this reason, in this thesis, we propose new non-random methods for building nested dichotomies. These new methods follow the idea of separating in the higher levels those classes that are easier to separate; and, in the lower levels those classes that are more difficult to separate. This is because if there were a mistake in some level, this mistake cannot be corrected in later levels; therefore this object will be misclassified.
In order to evaluate the performance of the proposed methods for building nested dichotomies, their accuracy results, over some datasets from the UCI repository, were compared against the results obtained using methods that randomly build nested dichotomies.

Índice general

1. Introducción **1**

1.1. Descripción del problema 1

1.2. Objetivos 3

1.2.1. Objetivo general 3

1.2.2. Objetivos específicos 4

1.3. Organización de la tesis 4

2. Marco teórico **5**

2.1. Conceptos preliminares 5

2.2. Funciones de Comparación 7

2.3. Métodos de Binarización 9

2.4. Ensambles 15

2.5. Validación 16

2.6. Significancia Estadística 17

3. Trabajo relacionado **19**

4. Métodos propuestos **27**

4.1. Método DAA 27

4.2. Método DAAR 31
4.3. Método DAAP 34
4.4. Resultados experimentales 35
4.5. Experimentos con los método DAA, DAAR y DAAP 36

5. Conclusiones **51**

5.1. Sumario 51
5.2. Conclusiones 52
5.3. Aportación 53
5.4. Trabajo Futuro 53

Índice de figuras

2.1. Esquema de la clasificación no supervisada. 6
2.2. Esquema de la clasificación supervisada. 7
2.3. Ejemplo de métodos de binarización de clases, para un problema de 4 clases a) OVO b) OVA c) Dicotomía anidada. 11
2.4. Diferentes dicotomías anidadas para un problema de 4 clases. . . 13
2.5. Ejemplo de una validación cruzada de 3 pliegues. 17

4.1. Método DAA para construir dicotomías anidadas. 29
4.2. Esquema de la fase de creación del la dicotomía anidada propuesta, para un ejemplo de 4 clases. 30
4.3. Representación de la distancia entre dos clases considerando sus radios. 32
4.4. Método DAAR para construir dicotomías anidadas basadas en agrupamientos y radios. 33
4.5. Ejemplo del problema de la obtención del radio de una clase como la distancia a la instancia más lejana. 34

Índice de tablas

2.1. Número de dicotomías anidadas que pueden formarse dependiendo del número de clases que tenga la base de datos. 14

4.1. Características de las bases de datos utilizadas en los experimentos. 37
4.2. Resultados en promedio de la precisión obtenida por cada método, de acuerdo al clasificador base utilizado. 38
4.3. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo Bagging. 39
4.4. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo AdaBoost. 39
4.5. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo MultiBoost. 40
4.6. Comparación método a método con el clasificador base C4.5. . . . 41
4.7. Comparación método a método con el clasificador base *Random Forest*. 41
4.8. Comparación método a método con el clasificador base 1-NN. . . 42
4.9. Comparación método a método con el clasificador base 3-NN . . . 42
4.10. Comparación método a método con el clasificador base *Naive Bayes*. 43
4.11. Comparación método a método Bagging con C4.5. 44

4.12. Comparación método a método Bagging con Random Forest . . . 44
4.13. Comparación método a método Bagging con 1-NN 44
4.14. Comparación método a método Bagging con 3-NN 45
4.15. Comparación método a método Bagging con Naive Bayes 45
4.17. Comparación método a método AdaBoost con Random Forest . . 46
4.18. Comparación método a método AdaBoost con 1NN 46
4.16. Comparación método a método AdaBoost con C4.5 46
4.19. Comparación método a método AdaBoost con 3NN 47
4.20. Comparación método a método AdaBoost con Naive Bayes 47
4.22. Comparación método a método Multiboost con Random Forest . 48
4.23. Comparación método a método Multiboost con 1-NN 48
4.21. Comparación método a método Multiboost con C4.5 48
4.24. Comparación método a método Multiboost con 3-NN 49
4.25. Comparación método a método Multiboost con Naive Bayes . . . 49

Capítulo 1

Introducción

En este capítulo se describe el problema a resolver en el marco de esta tesis, se presentan los objetivos de la investigación y se describe la organización de este documento.

1.1. Descripción del problema

La clasificación supervisada tiene como objetivo crear, utilizando objetos previamente clasificados (conjunto de entrenamiento), un modelo capaz de clasificar nuevos objetos. La mayoría de los métodos de clasificación supervisada fueron originalmente planteados para resolver problemas de dos clases (clasificación binaria), sin embargo en la práctica hay problemas que involucran más de dos clases. La clasificación multiclase se refiere a la clasificación de objetos en tres o más clases, en donde cada objeto pertenece a una sola clase. Este problema se ha tratado de resolver empleando clasificadores multiclase, sin embargo el problema de clasificación se vuelve más complejo cuando se tienen más clases, ya que el clasificador tendrá que considerar todas las clases para tomar decisión, con lo cual aumenta la

posibilidad de cometer errores. Por este motivo, para tratar con este problema, se han propuesto métodos que descomponen un problema multiclase en varios problemas de clasificación binaria (solo dos clases), con el objetivo de simplificar el problema y de este modo tener mejores resultados en la calidad de la clasificación. Ejemplos de estos métodos, llamados métodos de binarización de clases, son: uno-contra-uno [Debnath et al., 2004] y uno-contra-todos [Rifkin and Klautau, 2004].

Una dicotomía anidada es un método de binarización de clases en el cual un problema de multiclasificación se divide en varios problemas de clasificación de dos clases con la ayuda de un árbol binario, en donde la raíz del árbol contiene el conjunto de todas las clases. Después las clases son divididas en dos conjuntos de clases llamadas superclases y se crea un modelo para distinguir entre éstas. Este proceso se repite dividiendo cada superclase hasta que las superclases contengan una sola clase de las clases originales. Para clasificar un nuevo objeto se recorre el árbol desde la raíz, usando los modelos binarios para seleccionar la rama que se seguirá. Cuando se llega a una hoja se asigna al nuevo objeto la clase asociada a esta hoja.

Para cada problema multiclase se pueden construir diferentes dicotomías anidadas. Encontrar una buena dicotomía anidada ayudaría a reducir el error producido por el clasificador. Existen diversas maneras de construir estas dicotomías anidadas [Dong et al., 2005, Frank and Kramer, 2004]. Sin embargo, en la mayoría de estos métodos, la separación de las clases se hace de forma aleatoria, lo cual no garantiza que se encuentre una buena dicotomía anidada que ayude a mejorar la clasificación.

Por este motivo, en este trabajo de tesis se propone desarrollar un método no aleatorio para construir un árbol de dicotomías anidadas que separe las clases más fáciles de distinguir en los niveles superiores y en los niveles inferiores las clases más difíciles. Con esto se busca cometer menos errores en los niveles superiores, lo cual debería redundar en una mejor clasificación. Esto es importante pues en un árbol de dicotomías anidadas si se comete un error, en algún nivel, éste no puede ser corregido en los niveles inferiores y por lo tanto se convierte en un error de clasificación en el resultado final.

1.2. Objetivos

Dado que no existen métodos para construir dicotomías anidadas que no involucren aleatoriedad, en esta tesis se pretende proponer un nuevo método no aleatorio para construirlas. Por este motivo, los objetivos de la tesis son los siguientes.

1.2.1. Objetivo general

Desarrollar un método basado en agrupamiento para construir un árbol de dicotomías anidadas para la clasificación multiclase, que obtenga mejor precisión de clasificación que los métodos reportados en la literatura que construyen dicotomías aleatoriamente.

1.2.2. Objetivos específicos

1. Encontrar una manera de determinar cuándo dos grupos de clases son más fáciles de separar.

2. Definir un método para construir un árbol de dicotomías que coloque en los niveles superiores del árbol las clases más fáciles de separar y en los niveles inferiores las clases más difíciles.

3. Construir ensambles de dicotomías que permitan mejorar la precisión de la clasificación.

Las principales contribuciónes de esta tesis son 3 nuevos métodos no aleatorios para construir dicotomías anidadas.

1.3. Organización de la tesis

El resto del documento de tesis está organizado de la siguiente manera. Primero, en el capítulo 2, se darán los conceptos básicos necesarios para entender el resto de la tesis. En el capítulo 3 se describirá el trabajo relacionado con la presente investigación. En el capítulo 4 se introducirán los nuevos métodos propuestos para la construcción de dicotomías y se mostrarán los resultados experimentales obtenidos al comparar los métodos propuestos contra otros métodos de construcción de dicotomías. En el capítulo 5 se expondrán nuestras conclusiones y algunas direcciones para trabajo futuro.

Capítulo 2

Marco teórico

Con el objetivo de familarizar al lector con los términos empleados a lo largo de este documento, en este capítulo se introduce una serie de conceptos básicos necesarios para entender el resto de la tesis.

2.1. Conceptos preliminares

El reconocimiento de patrones es la ciencia que se ocupa de los procesos sobre ingeniería, computación y matemáticas relacionados con objetos físicos y/o abstractos, con el propósito de extraer información que permita establecer propiedades de o entre conjuntos de dichos objetos. En esta tesis se asume que los objetos son descritos por un conjunto de atributos.
Entre los principales problemas abordados en el área de Reconocimiento de Patrones tenemos:

1. Clasificación no supervisada. En la clasificación no supervisada no se tiene conocimiento de las clases a las cuales pertenecen las instancias de una muestra, únicamente se conocen las descripciones de las instancias. En este tipo de clasificación, el objetivo consiste en encontrar las clases en las que se agrupan las instancias. En la figura 2.1 se muestra un esquema de la clasificación no supervisada.

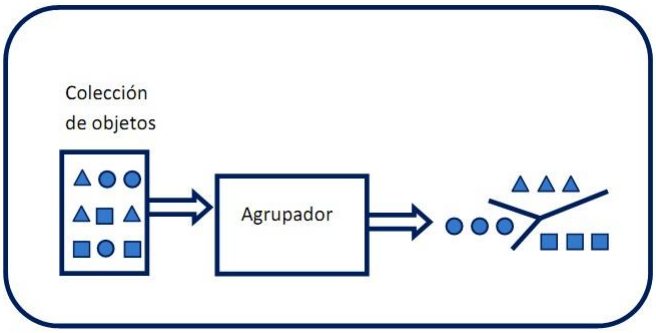

Figura 2.1. Esquema de la clasificación no supervisada.

2. Clasificación supervisada. En la clasificación supervisada, dado un universo dividido en clases, se cuenta con una colección de instancias para los cuales, además de su descripción, se conoce la clase a la que pertenece cada instancia. A la colección de instancias ya clasificados también se le conoce como conjunto de entrenamiento, ya que mediante este conjunto, el clasificador se "entrena" para realizar futuras clasificaciones. El problema de clasificación supervisada consiste en asignar a las nuevos instancias (no clasificadas), una clase con base en los casos conocidos. En la figura 2.2 se muestra un esquema de la clasificación supervisada

3. Selección de Variables. Consiste en seleccionar, entre las características o atributos que se les puede medir a las instancias, aquellas que sean más útiles

para describir al conjunto de entrenamiento, dependiendo del problema a resolver.

En este trabajo de tesis se abordará el problema de la clasificación supervisada en problemas multiclase.

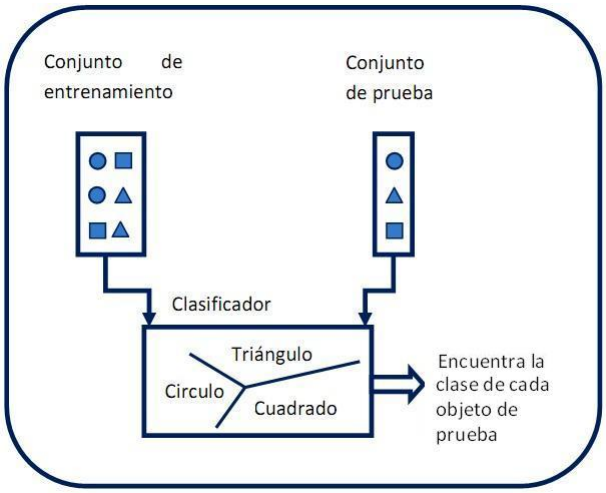

Figura 2.2. Esquema de la clasificación supervisada.

2.2. Funciones de Comparación

Dada una colección de instancias T, donde las instancias se describen mediante un conjunto de atributos $\{x_1, x_2, ..., x_n\}$. Cada atributo es definido dentro de un dominio $D_i = dom(x_i)$; $i = 1, ..., n$. El dominio de un atributo es el conjunto de todos los valores que puede tomar el atributo x_i. Se tienen diferentes tipos de atributos en dependencia del dominio. Por ejemplo los atributos pueden ser Booleanos $D(x_i) = \{0, 1\}$, números enteros o reales $D(x_i) \subseteq \mathbb{R}$, ó valores categóricos, por ejemplo el dolor *D(dolor)* = {*nada, poco, regular, mucho*} , así

$D(x_i) = \{k_1, k_2, ..., k_m\}$ donde k representa un valor categórico y m representa en número total de posibles valores.

De acuerdo con la descripción de cada instancia, muchos algoritmos calculan, mediante una función de comparación, qué tan cercano se encuentra una instancia de otra o qué tan parecida es una instancia con relación a otra. Entre las distancias (funciones de comparación) más comunes se encuentran:

Distancia Euclidiana: Sean p y q dos instancias descritas con n atributos; $p = (p_1, p_2, ..., p_n)$ y $q = (q_1, q_2, ..., q_n)$. La distancia Euclidiana entre p y q se define como:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2}$$

Distancia Manhattan: Dados p y q como antes, la distancia Manhattan entre p y q se define como sigue:

$$d(p, q) = \sum_{i=1}^{n} \mid p_i - q_i \mid$$

Estas distancias están definidas para atributos numéricos, pero en muchos problemas de clasificación se tienen atributos no numéricos. Las siguientes funciones permiten comparar instancias descritas por atributos numéricos y no numéricos (datos mezclados).

La función de comparación HVDM entre dos instancias p y q, como antes, se define de la siguiente manera:

$$HVDM(p, q) = \sqrt{\sum_{i=1}^{d} F_i(p_i, q_i)} \tag{2.1}$$

Donde $F_i(p_i, q_i)$ se define como:

$$F_i(p_i, q_i) = \begin{cases} overlap(p_i, q_i) & \text{Si } x_i \text{ es no numérico} \\ \frac{|p_i - q_i|}{max_i - min_i} & \text{Si } x_i \text{ es numérico} \end{cases} \tag{2.2}$$

Donde max_i y min_i son los valores máximo y mínimo que puede tomar el atributo x_i. Ademas:

$$overlap(p_i, q_i) = \begin{cases} 0 & \text{Si } p_i = q_i \\ 1 & \text{En otro caso} \end{cases} \tag{2.3}$$

En este trabajo de tesis se utilizará para comparar instancias la función HVDM porque permite trabajar con descripciones de instancias con atributos numéricos y no numéricos (datos mezclados).

2.3. Métodos de Binarización

Muchos de los clasificadores supervisados han sido diseñados para problemas de dos clases, a este tipo de clasificadores se les conoce como clasificadores binarios, posteriormente estos métodos se han extendido para resolver problemas de clasificación con más clases (clasificación multiclase). Sin embargo, el problema se complica conforme crece el número de clases, es por eso que se han propuesto métodos de binarización de clases, los cuales consisten en descomponer un problema de múltiples clases en varios problemas de dos clases y resolverlos mediante clasificadores binarios [[Fürnkranz, 2002], [Garcia-Pedrajas and Ortiz-Boyer, 2006], [Quost et al., 2007a] y [Huhn and Hüllermeier, 2008]]. A este problema se le conoce como binarización de clases. Existen diferentes maneras de abordar este problema, pero los más ampliamente estudiados son los métodos uno-contra-uno

(OVO por sus siglas en inglés One Vs One) y uno-contra-todos (OVA por sus siglas en inglés One Vs All).

El método OVO consiste en generar todas las posibles combinaciones de dos clases y construir un clasificador binario para cada combinación. Así un problema de clasificación con m clases se descompone en $m(m-1)/2$ problemas binarios. Para clasificar una instancia nueva, se evalúan todos los clasificadores binarios y se toma la decisión final con base en los resultados de los clasificadores binarios, comúnmente por votación.

El método OVA consiste en transformar el problema de clasificación supervisada con m clases en m problemas binarios cada uno separando una clase de las demás, así el problema de m clases se resuelve con m clasificadores binarios. Para clasificar una nueva instancia éste se pasa a los m clasificadores binarios y se toma la decisión combinando los resultados de los m clasificadores binarios, comúnmente seleccionando aquel que seleccione una clase individual con la mayor evidencia.

Otro método de binarización de clases consiste en la construcción de dicotomías anidadas (ND por sus siglas en inglés Nested Dichotomies). Una dicotomía anidada consiste en separar un conjunto de clases en dos subconjuntos de tal manera que cada uno es excluyente del otro y construir un clasificador binario para separar estos conjuntos de clases. Este proceso se repite recursivamente para cada subconjunto, hasta separar clases individuales. Una dicotomía anidada es un árbol binario que cumple las siguientes propiedades:

- Cada nodo tiene asociado un conjunto no vacío de clases.
- El nodo raíz incluye todas las clases, mientras que los nodos hoja tienen una sola clase.

- El árbol es estrictamente binario, es decir, todos los nodos que no son nodos hoja tienen exactamente dos hijos.
- Las clases de dos nodos hermanos forman una partición de las clases en el nodo padre, es decir, su intersección es vacía y su unión es el conjunto de todas las clases del nodo padre.
- Cada nodo interno tiene asociado un clasificador binario, que discrimina entre los dos conjuntos de clases asociadas con los nodos hijo.

En la figura 2.3 se muestra un ejemplo de cada uno de estos métodos de binarización, aplicados a un problema de cuatro clases.

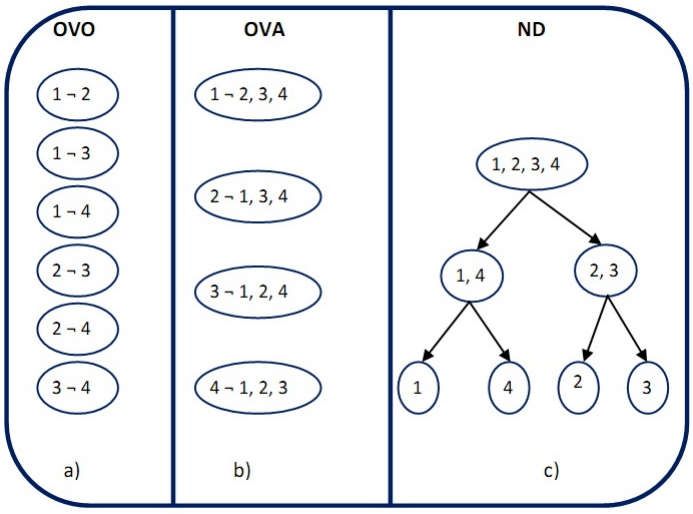

Figura 2.3. Ejemplo de métodos de binarización de clases, para un problema de 4 clases a) OVO b) OVA c) Dicotomía anidada.

El problema con las dicotomías anidadas es que existen muchas posibles estructuras de árbol que se pueden formar y no se tiene conocimiento previo acerca

de si una dicotomía anidada es más apropiada que otra. En la figura 2.4 se muestra un ejemplo de diferentes dicotomías anidadas para un mismo problema. El problema de tener diferentes dicotomías anidadas es que a menudo conducen a resultados de clasificación diferentes, porque los clasificadores binarios para cada nodo tratan diferentes problemas de dos clases. El número total de dicotomías anidadas que pueden formarse a partir de un problema de n clases está dado por la siguiente función de recurrencia:

$$T(1) = 1$$
$$T(n) = (2n-3) \times T(n-1)$$

Se puede probar por inducción que toda dicotomía anidada de k clases tiene k nodos hoja y $(k-1)$ nodos internos es decir, sumando $k + (k-1)$ en total una dicotomía tiene $2k-1$ nodos. Entonces una dicotomía anidada de $n-1$ clases tendrá $2(n-1)-1 = 2n-3$ nodos. Por lo tanto el número de dicotomías de tamaño n es igual al número de dicotomías de tamaño $n-1$ ($T(n-1)$) multiplicado por $2(n-1)-1 = 2n-3$ nodos en cada una y la nueva clase se puede agregar como hermano (agregando el nodo padre correspondiente) en cualquiera de ellos, por lo tanto el número de dicotomías anidadas de tamaño n es $(2n-3) \times T(n-1)$.

Para tratar de reducir el número de dicotomías que pueden formarse, en [Dong et al., 2005] consideran solo las dicotomías que se pueden formar con árboles balanceados, esto además garantiza que la profundidad del árbol sea logarítmica. En [Dong et al., 2005] encuentran que la relación de recurrencia que define el número de posibles árboles balanceados de dicotomías es:

$$T(c) = \begin{cases} \frac{1}{2}\binom{c}{c/2} T\left(\frac{c}{2}\right) T\left(\frac{c}{2}\right) & \text{Si } c \text{ es par} \\ \binom{C}{(C+1)/2} T\left(\frac{c+1}{2}\right) T\left(\frac{c-1}{2}\right) & \text{Si } c \text{ es impar} \end{cases} \tag{2.4}$$

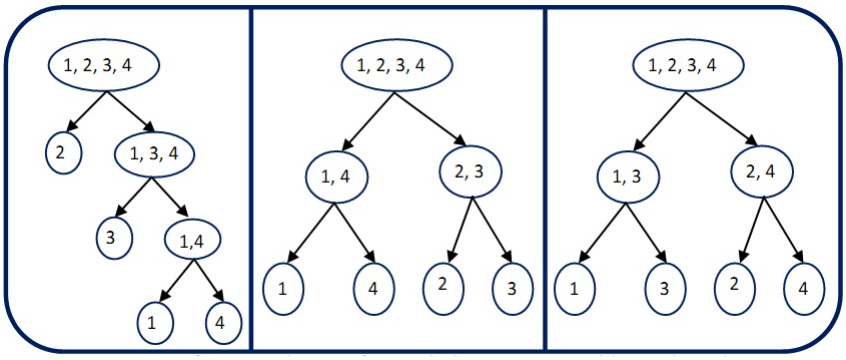

Figura 2.4. Diferentes dicotomías anidadas para un problema de 4 clases.

Donde $T(1) = 1$ y $T(2) = 1$, y c es el número de clases. En la tabla 2.1 se muestra el número de dicotomías anidadas y dicotomías balanceadas que pueden formarse dependiendo del número de clases que tenga la base de datos. Puede notarse que en el caso de dicotomías balanceadas se reduce significativamente el número de posibles dicotomías anidadas. A pesar de reducir el número de dicotomías para un problema de n clases, esto no garantiza que se obtenga una buena partición que ayude a mejorar la clasificación.

Actualmente los métodos de construcción de dicotomías anidadas construyen el árbol aleatoriamente lo cual no garantiza encontrar una buena dicotomía. Todo esto nos motiva a desarrollar métodos para construir dicotomías anidadas no aleatorias que ayuden a mejorar la calidad de la clasificación.

Para las pruebas que se realizaron se consideraron los siguientes clasificadores base (binarios) que están implementados en weka [Holmes et al., 1994].

- C4.5. Este clasificador construye un árbol de decisión.

Tabla 2.1. Número de dicotomías anidadas que pueden formarse dependiendo del número de clases que tenga la base de datos.

Número de clases	Número de dicotomías anidadas	Número de dicotomías anidadas balanceadas
2	1	1
3	3	3
4	15	3
5	105	30
6	945	90
7	10,395	315
8	135,135	315
9	2,027,025	11,340
10	34,459,425	113,400
11	654,729,075	1,247,400
12	13,749,310,575	3,742,200

- K-NN. Es un clasificador en el cual la instancia a clasificar se clasifica de acuerdo a las k instancias del conjunto de entrenamiento que se encuentren más cercanas a la nueva instancia.

- *Random Forest*: Es un ensamble tipo *bagging* que utiliza, como clasificadores base, árboles de decisión que se forman con un subconjunto aleatorio de atributos

- *Naive Bayes*: Es un clasificador basado en el teorma de Bayes, que asume independencia de atributos dada la clase.

Cabe mencionar que el método C4.5 se encuentra nombrado en weka como J48, y el método K-NN como ibk.

2.4. Ensambles

Los ensambles son métodos que combinan clasificadores. Estos métodos construyen un conjunto de clasificadores, y combinan de alguna manera (generalmente por votación) los resultados de varios clasificadores para clasificar nuevas instancias. La precisión obtenida por esta combinación de clasificadores generalmente supera la precisión de cada clasificador individual [Ahn et al., 2007].

Cada clasificador puede verse como un experto, por lo que es posible que se produzcan predicciones más confiables si se combinan las predicciones de varios clasificadores. Dentro de los tipos de ensamble más comunes se encuentran:

Bagging (Bootstrap Aggregating) [Breiman, 1996]. En este método, los clasificadores se forman construyendo cada clasificador con diferentes instancias del conjunto de entrenamiento; por cada clasificador que se forma, se eligen muestras aleatorias con reemplazo, del conjunto de entrenamiento, es decir, se eligen n instancias (n <= número total de instancias en el conjunto de entrenamiento), de tal manera que cada instancia se elige de forma aleatoria sin retirarla del conjunto de entrenamiento, así cada instancia tiene la misma probabilidad de ser elegida cada vez que se selecciona una instancia. Posteriormente, se forman los clasificadores base con cada conjunto de instancias y para clasificar una nueva instancia se evalúan todos los clasificadores base y la clase más votada es la que se asigna.

Boosting [Freund and Schapire, 1997]. Este método consiste en crear un ensamble de clasificadores añadiendo un clasificador en cada paso. En el conjunto de entrenamiento cada instancia tiene asignado un peso. El clasificador que se añade en el paso k es entrenado a partir del conjunto de instancias original donde el peso de cada ejemplo ha sido modificado en la iteración $k - 1$. En cada iteración, entrena un modelo que minimiza la suma de los pesos de las instancias

mal clasificadas. Los errores en cada iteración se utilizan para actualizar los pesos de las instancias del conjunto de entrenamiento de manera que se incremete el peso de las instancias mal clasificadas. Generalmente se inicializa el peso de cada instancia con 1.

2.5. Validación

Estimar la certeza de un clasificador al clasificar es lo que comúnmente se conoce como validar, esto nos sirve para medir su capacidad de predicción y para saber qué tan confiable es el clasificador sobre nuevas instancias que se clasificarán en el futuro, la validación se realiza basándose en la tasa de error del clasificador, entendiendo el error como la clasificación incorrecta de una instancia.

La validación simple consiste en dividir en dos conjuntos complementarios los datos de la muestra, uno se utiliza para construir el modelo y el otro para prueba. Esta división de datos generalmente es aleatoria.

La validación cruzada consiste en aplicar k veces la validación simple, la muestra se divide en k conjuntos, a cada partición se le llama pliegue y por cada partición se construye el modelo con las $k-1$ particiones restantes como entrenamiento y se evalúa el modelo construido con la partición seleccionada. Se calcula cuántos elementos fueron correctamente clasificados para cada partición dividida por el tamaño de la partición (*accuracy*); al final se promedian los resultados. En la figura 2.5 se muestra gráficamente cuáles son las particiones que se utilizan en el conjunto de entrenamiento y el conjunto de prueba para una validación de 3 pliegues ($k = 3$).

El máximo valor que puede tomar k es el número de instancias de la muestra

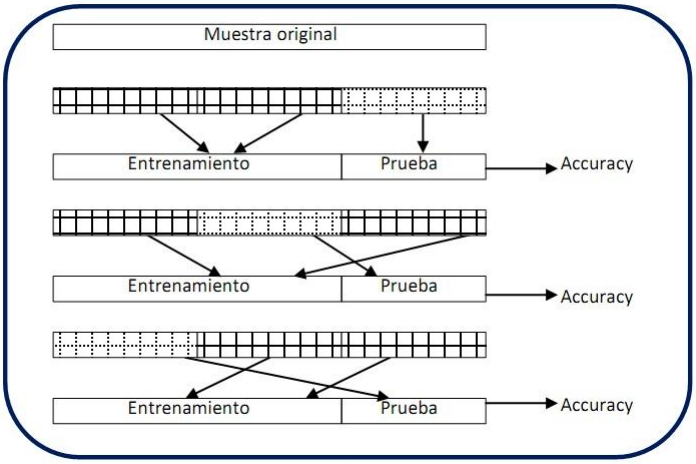

Figura 2.5. Ejemplo de una validación cruzada de 3 pliegues.

original. Usualmente se toma $k = 10$, si la muestra es muy grande $k = 3$, si la muestra es muy pequeña se toma el máximo valor de k, es decir, $k =$ número de instancias en la muestra.

Otra técnica muy utilizada es *Bootstrap*, la cual consiste en elegir los elementos de forma aleatoria con reemplazo, es decir, que una instancia puede ser seleccionado más de una vez, y construir un conjunto de entrenamiento de N instancias. Luego se forma el conjunto de prueba con las instancias que no se hayan seleccionado. Este proceso se repite k veces y al final se promedian los resultados de todas las pruebas.

2.6. Significancia Estadística

La significancia estadística se utiliza para comparar los resultados de un clasificador contra los resultados de otro. Su objetivo es validar que los resultados obtenidos no son debidos al azar. Se busca decidir si ambos clasificadores obtienen

resultados equivalentes (son competitivos uno con otro) o si uno es mejor que el otro.

Se necesita la tabla de las precisiones de cada método evaluado por cada clasificador sobre los mismos datos, comúnmente estas pruebas se hacen sobre la validación cruzada. Usualmente la hipótesis que se prueba es que los resultados son equivalentes, si se rechaza la hipótesis se infiere que el clasificador que obtiene mejores resultados es el mejor y que este resultado no se debe al azar, es decir, la diferencia es estadísticamente significativa.

Para los experimentos se realizará la prueba Wilcoxon, esta prueba es no paramétrica, es decir, no se requiere la condición de distribución normal y permite establecer la hipótesis sobre la diferencia de medias de los resultados de clasificación de dos clasificadores independientes. Esta prueba se basa en el número de veces que los valores de los resultados obtenidos con una muestra son excedidos por los valores de los resultados obtenidos con la otra muestra y esto sucede en términos de los rangos. Los datos se ordenan conjuntamente del menor al mayor, como si provinieran de una misma muestra, y se le asigna a cada uno de ellos el rango, entendiendo éste como el lugar que ocupan dentro del ordenamiento. Se establecen las diferencias de magnitudes entre las observaciones, pudiendo ser estas (+ y -), y de acuerdo a un umbral se determina si una prueba es igual a la otra dando como resultado 0, ó se rechaza esta hipótesis y se determina si un clasificador es significativamente mejor que el otro [Demšar, 2006].

Capítulo 3

Trabajo relacionado

En este capítulo se describen los métodos más recientes para construir dicotomías anidadas, que permitan resolver problemas de clasificación multiclase.

Muchos problemas de clasificación del mundo real son problemas multiclase, es decir, se tienen más de dos clases de objetos. Existen dos enfoques para tratar este tipo de problemas. El primero es adaptar un clasificador binario para predecir múltiples clases directamente, es decir, crear un clasificador multiclase. El otro enfoque es crear varios problemas de dos clases para hacer predicciones multiclase basadas en las predicciones obtenidas de estos problemas de dos clases. En este último enfoque se han propuesto algunos trabajos como *error-correcting output codes* [Allwein et al., 2001] clasificación por parejas [Quost et al., 2007b], etc. Hay dos estrategias principales para abordar los problemas multiclase con clasificadores binarios. La primera consiste en construir un clasificador binario para distinguir cada clase de las demás, a este método se le llama uno-contra-todos [Rifkin and Klautau, 2004]. La segunda estrategia, llamada uno-contra-uno, consiste en crear un clasificador binario por cada par de clases

[Hastie and Tibshirani, 1998]. Otra estrategia menos estudiada, para trabajar con clasificadores binarios para resolver problemas multiclase es construir una dicotomía anidada, que consiste en crear un árbol binario que en la raíz separe las clases en dos grupos de clases (superclases) y crear un clasificador binario, que separe estos dos grupos, cada grupo es recursivamente subdividido en dos grupos hasta separar todas las clases.

En [Aoki and Kudo, 2002] proponen un método para separar clases de un problema multiclase, el método consiste en construir un árbol de decisión, en donde en cada nivel realizan una selección de atributos y separan las clases que pueden ser divididas, colocando una sola clase en cada hoja del árbol, el método termina cuando cada clase se encuentra en una hoja del árbol. Para clasificar una instancia, se recorre el árbol como un árbol de decisión y cuando se llega a un nodo hoja, se le asigna la clase del nodo hoja. Los autores realizaron pruebas con bases de datos artificiales y muestran que su método ofrece la ventaja de separar clases que muestran dependencias de otras clases.

En [Frank and Kramer, 2004] abordan el problema de clasificación multiclase utilizando dicotomías anidadas. Para representar la dicotomía anidada utilizan un árbol binario que recursivamente divide el conjunto de clases en dicotomías, esto lo hacen de la siguiente manera:
En cada nodo interno A, incluyendo la raíz, se divide el conjunto de clases A asociadas con este nodo dentro de dos subconjuntos B y C que son mutuamente excluyentes, y su unión contiene todas las clases de A. Para esto, cada clase de A se coloca aleatoriamente en B o en C. La raíz del árbol contiene todas las clases del problema multiclase y cada hoja contiene una sola clase, es decir, para un problema de n-clases se tienen n nodos hoja. Para construir los clasificadores

binarios, se toman solo las instancias que pertenecen a las clases de cada nodo y por cada nodo interno A se crea el clasificador binario que separa las clases del conjunto B de las clases del conjunto C. En este trabajo utilizaron como clasificadores binarios a C4.5 y regresión logística. Como se explicó en el capítulo 2 existen muchas dicotomías anidadas diferentes que pueden formarse para un problema de n clases ($n > 2$), los resultados de clasificación obtenidos por diferentes dicotomías anidadas puede variar, ya que cada dicotomía anidada contiene diferentes clasificadores binarios, eso significa que se resuelven problemas binarios diferentes (con diferentes formas de separar las clases involucradas). Los autores afirman que no se tienen razones a priori para afirmar que una dicotomía sea preferible para un problema multiclase en particular, por lo tanto no hay razón para creer que una dicotomía es mejor que las otras. Por consecuencia, esto hace que todas las dicotomías anidadas sean tratadas de igual manera, eligiendo, en cada nodo, una dicotomía de forma aleatoria.

Además, proponen realizar un ensamble de dicotomías anidadas (END), generando aleatoriamente las dicotomías anidadas, es decir, generan árboles de dicotomías anidadas y los usan como clasificadores del ensamble. Para determinar la clase de una instancia, evalúan todas las dicotomías anidadas y la clase con más votos es la que se le asigna a esa instancia. En tiempo de ejecución el método END está limitado por la profundidad del árbol formado por cada dicotomía anidada, en el peor de los casos podría formar un árbol en donde solo separe una clase por cada nodo, y esto sería similar al método uno-contra-todos. Los autores realizaron experimentos con bases de datos del repositorio UCI [Asuncion and Newman, 2007], mostrando que sus resultados de clasificación con ENDs son mejores que aplicar métodos de binarización [Hastie and Tibshirani, 1998], y *error correcting output codes*.

En [Dong et al., 2005] tratan el problema de clasificación multiclase con dicotomías anidadas realizando un ensamble de dicotomías balanceadas. Dado que existe un gran número de dicotomías anidadas para resolver un problema de n clases, proponen trabajar solo con aquellas dicotomías que sean balanceadas, reduciendo así el espacio de las dicotomías anidadas. Para formar la dicotomía anidada balanceada, en cada nodo se reparten aleatoriamente las clases, teniendo cuidado de equilibrar el número de clases en los nodos hijos. Luego construyen un ensamble con dicotomías anidadas balanceadas y lo llaman *"ensemble of class-balanced nested dichotomies"* (ECBND). Además proponen una variante que consiste en balancear los datos, ya que a pesar de tener dicotomías anidadas balanceadas, algunos problemas llegan a formar árboles desbalanceados en cuanto al número de objetos por nodo. Para formar la dicotomía anidada con los datos balanceados, asignan aleatoriamente clases a dos subconjuntos hasta que el tamaño de los datos de entrenamiento en uno de los subconjuntos supere la mitad de la cantidad total de datos de entrenamiento en el nodo padre asignando el resto de los datos al otro nodo. Con esta variante conservan la aleatoriedad en la asignación de las clases para cada dicotomía, para preservar la diversidad de dicotomías anidadas en el comité generado en el ensamble. Finalmente, forman un ensamble y lo llaman *"ensemble of data-balanced nested dichotomies"* (EDBND). Realizan experimentos con bases de datos del repositorio UCI y se comparan con ENDs, mostrando ser mejores en tiempo de entrenamiento, sin embargo no lograron obtener mejores resultados en la precisión.

En [Rodríguez et al., 2010] consideran que los métodos de ensamble frecuentemente generan mejores resultados de clasificación que los clasificadores individuales. Dado que las dicotomías anidadas pueden ser utilizadas con árboles de

decisión como clasificadores base, una dicotomía anidada de árboles de decisión puede ser el clasificador base para un ensamble. Este enfoque es llamado Bosque de dicotomías anidadas (FND). Consideran las siguientes formas de hacer ensambles:

Bagging [Breiman, 1996] Los datos de entrenamiento de cada clasificador se obtienen de muestras aleatorias con reemplazo, del conjunto de entrenamiento original. El tamaño de la muestra es del mismo tamaño de la muestra original.
Random Forest [Statistics and Breiman, 2001]. Utilizan *Bagging* con árboles de decisión aleatorios. Estos árboles seleccionan aleatoriamente un atributo por nodo y solo un subconjunto de atributos es considerado.
Random Subspaces [Ho, 1998] Cada clasificador es entrenado usando todas las instancias del conjunto de entrenamiento, pero con un subconjunto aleatorio de atributos.

AdaBoost [Freund and Schapire, 1997] Considera un método *boosting*. Cada instancia del conjunto de entrenamiento tiene un peso, el clasificador base debe considerar estos pesos y si una instancia es mal clasificada, su peso se incrementa, y si se clasifica correctamente, entonces su peso se decrementa.

MultiBoost [Webb, 2000] Es un método que combina las ideas de *boosting* y *bagging*. Trabaja similar a *boosting*, pero después de un número de iteraciones el peso de las instancias es reasignado aleatoriamente.
En los experimentos probaron a las siguientes configuraciones para sus ensambles:
F: Es un ensamble tipo *bagging* de árboles de decisión.
END: Es un ensamble de dicotomías anidadas con árboles de decisión como clasificadores base.

ENDF: Ensamble de dicotomías anidadas usando *Random Forest*.
FND: Utiliza un método *Random Forest* con dicotomías anidadas de árboles de decisión como clasificador base.
En [Rodríguez et al., 2010] se realizaron experimentos con los diferentes tipos de ensamble, cada ensamble contenía 10 clasificadores base y en el caso del número de dicotomías en cada bosque también fue de 10. Utilizaron bases de datos del repositorio UCI, y de acuerdo a sus experimentos muestran que la mejor manera de realizar ensambles con dicotomías anidadas es con el método END con el ensamble de tipo *MultiBoost*.

En [Aoki and Kudo, 2010] proponen un método para crear la dicotomía separando una clase en cada nivel de la misma. La selección de la clase a separar la hacen seleccionando los atributos que ayuden a distinguir una clase de las demás. Para cuantificar en qué medida una clase se distingue de las demás, los autores utilizan la precisión obtenida por un clasificador lineal, un clasificador cuadrático, 1-NN, 3-NN y una máquina de vectores de soporte (SVM), Evalúan estos 5 clasificadores y seleccionan al que mejor clasifique en cada nodo. El método termina cuando todas las clases son separadas. Realizarón experimentos con bases de datos de 4 y 9 clases del repositorio UCI, y crearon una base de datos artificial, y de acuerdo a sus resultados obtienen mejores resultados de precisión cuando trabajan con un gran número de clases, utilizando como clasificador base un árbol de decisión.

Como se puede observar, las dicotomías anidadas son una buena alternativa para resolver problemas multiclase, ya que reducen en cada paso el problema multiclase en un problema binario, esto es muy útil cuando se tienen problemas

con un gran número de clases. En [Aoki and Kudo, 2002, Aoki and Kudo, 2010] se emplean diferentes técnicas para separar las clases pero estos métodos resultan costosos en tiempo de ejecución, por la selección de atributos que realizan en cada nivel del árbol y en [Aoki and Kudo, 2010] el método propuesto resulta aún más lento ya que se requieren evaluar los 5 clasificadores en cada nodo de la dicotomía para seleccionar el clasificador que mejor convenga. Sin embargo, no logran buenos resultados de calidad.

En los trabajos [Frank and Kramer, 2004, Dong et al., 2005, Rodríguez et al., 2010] muestran que los ensambles de dicotomías anidadas mejoran los resultados de clasificación, sin embargo los métodos propuestos para la construcción de dicotomías anidadas son aleatorios. Esto nos motiva a tratar de encontrar un nuevo método para construir dicotomías de forma no aleatoria que separen las clases más fáciles de separar en los primeros niveles de la dicotomía anidada y en los niveles inferiores separen las clases más difíciles, generando así un método de construcción de dicotomías anidadas que ayude a mejorar la precisión de la clasificación.

Capítulo 4

Métodos propuestos

En este capítulo se introducen los métodos propuestos para crear dicotomías anidadas que ayuden a resolver problemas multiclase transformándolos en varios problemas binarios (2 clases). Inicialmente introduciremos las ideas subyacentes en los métodos propuestos así como una descripción de los mismos. Posteriormente mostraremos una serie de experimentos para mostrar el desempeño de los métodos propuestos y compararemos sus resultados con los métodos que construyen dicotomías anidadas aleatoriamente ND de [Frank and Kramer, 2004], ND(*Data-Balanced*) y ND(*Class-Balanced*) de [Dong et al., 2005].

4.1. Método DAA

En una dicotomía anidada, cuando alguno de los clasificadores binarios comete un error, este error se transmite a los niveles inferiores donde no puede ser corregido. Por este motivo, es preferible que en los niveles superiores se comentan menos errores. Siguiendo esta idea, en esta tesis se propone un método para construir

dicotomías anidadas, que separe al principio las clases más fáciles de separar y en los niveles inferiores las clases más difíciles. Para medir la cercanía entre clases consideramos la media de la clase, ya que proporciona la información necesaria para determinar si una clase se encuentra más cercana de otra. La idea es que las clases más lejanas son las más fáciles de separar y por lo tanto deberían separarse en los primeros niveles. Para esto, agruparemos el conjunto de clases en dos grupos, dejando juntas las clases más cercanas y separadas las clases más lejanas. Para separar estos grupos seleccionaremos el par de medias más distantes, ya que representan a las clases más fáciles de separar. Posteriormente, agruparemos con cada una de estas medias el resto de las medias de las otras de clases, colocando cada media de la clase con la media más cercana (de entre las dos más distantes). En la figura 4.1 se muestra un ejemplo de cómo separar 4 clases, en este ejemplo se puede observar que la clase menos (-) y círculo (o), son muy parecidas y por eso son separadas hasta el último nivel de la dicotomía anidada.

El método propuesto que sigue esta idea lo llamamos Dicotomía Anidada basada en Agrupamiento (DAA). DAA separa en los primeros niveles las clases más fáciles de separar, es decir aquellas con las medias más lejanas, y posteriormente las más fáciles de separar, es decir, aquellas con las medias más cercanas. Este método consiste en 2 fases:

⋆ Fase 1. Construcción del clasificador. Se emplean todas las instancias del conjunto de entrenamiento para:

1. Elegir la media de cada clase, en caso de que los atributos sean numéricos. Si no lo son, elegir el elemento más parecido, en promedio, con las otras instancias de su clase.

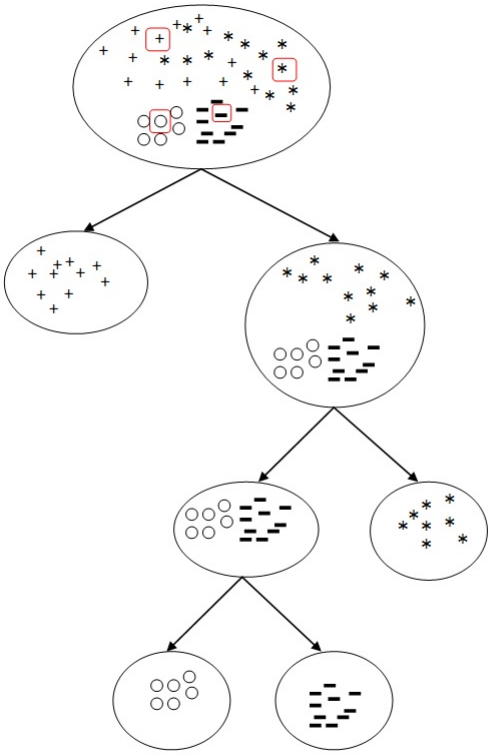

Figura 4.1. Método DAA para construir dicotomías anidadas.

2. Se crea la raíz del árbol con todas las clases y las instancias medias elegidas en el paso 1.

3. Creación de la Dicotomía (este proceso se realiza recursivamente hasta obtener una sola clase en cada hoja).

   a) Elegir las 2 medias menos parecidas (más lejanas), para formar los centros de los grupos

   b) Cada media del nodo padre, se agrupará con el centro más cercano, en caso de empates, se toma el primer grupo.

c) Se crea un nodo hijo para cada uno de los grupos y se repite el proceso recursivamente.

4. Cuando el árbol se ha construido, se entrenan los clasificadores binarios correspondientes en cada nodo interno del árbol. Las clases se forman tomando las instancias del conjunto de entrenamiento que están contenidas en las clases agrupadas en los nodos hijo.

⋆ Fase 2. Proceso de clasificación. Dada una instancia:

1. Se recorre el árbol desde la raíz, se avanza por la rama que el clasificador base seleccione, el recorrido continúa hasta llegar a una hoja.
2. Asignar la clase del nodo hoja, al que se llegó.

En la figura 4.2, se muestra un ejemplo de la fase 1 de construcción de la dicotomía anidada con el método DAA.

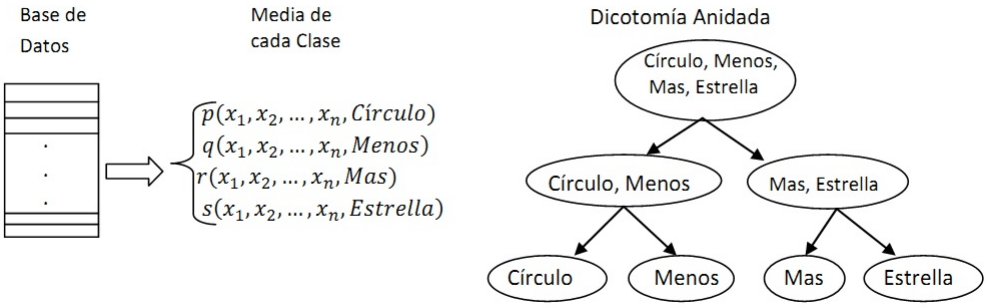

Figura 4.2. Esquema de la fase de creación del la dicotomía anidada propuesta, para un ejemplo de 4 clases.

4.2. Método DAAR

Un problema con el método DAA es que pueden existir clases cuyas medias están lejanas pero aún así exista mucho traslape entre ellas y por lo tanto sean difíciles de separar. Por otro lado, pueden existir clases cuyas medias estén cercanas pero no exista traslape entre ellas y por lo tanto sean fáciles de separar. Por este motivo se propuso una variante del método DAA que involucre el radio de cada clase al momento de calcular la cercanía entre clases. El radio de una clase es obtenido calculando la distancia entre su centroide y el elemento más lejano del centroide, en su misma clase. Para medir la cercanía de clases se propuso la fórmula 4.1, en donde C_1 y C_2 son las clases a comparar, M_1 y M_2 representan las medias de estas clases, d es la distancia entre medias (en esta tesis utilizaremos HVDM debido a que permite trabajar con atributros numéricos y no numéricos), $r1$ es el radio de la clase C_1 y $r2$ es el radio de la clase C_2.

$$D(C_1, C_2) = \frac{d(M_1, M_2)}{r_1 + r_2} \tag{4.1}$$

Si esta distancia vale 1, significa que las clases se encuentran una junto a otra y no hay traslape, si la distancia es mayor que 1 entonces significa que se encuentran separadas y si la distancia es menor que 1, significa que se tiene traslape entre estas clases. En la figura 4.3 se muestra de manera general la distancia entre dos clases considerando sus radios. En este ejemplo las clases 1 y 2 son más cercanas de acuerdo a la distancia propuesta debido a que existe traslape entre ellas. Por otro lado las clases 3 y 4 son más lejanas a pesar de que sus medias sean cercanas, debido a que no hay traslape entre ellas. Siguiendo esta idea, utilizaremos esta distancia para medir la cercanía entre clases y de este modo separaremos al principio de la dicotomía anidada las clases cuya distancia sea mayor, ya que

son las más fáciles de separar, y dejaremos en los niveles inferiores las clases con distancia menores ya que son las más difíciles de separar.

A este método lo llamamos Dicotomía Anidada basada en Agrupamiento y Ra-

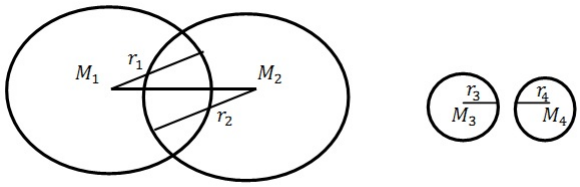

Figura 4.3. Representación de la distancia entre dos clases considerando sus radios.

dios (DAAR). En la figura 4.4 se muestra un ejemplo de la separación de clases, las medias de las clases Menos (-) y Círculo (o) se encuentran cerca, sin embargo sus elementos son más fáciles de separar, ya que los radios de las clases no se traslapan, por lo tanto son separadas y el resto de las clases se agrupan con la clase cuya distancia entre sus medias (utilizando la equación 4.1) sea la más cercana. La fase de construcción del clasificador para el método DAAR consiste en:

1. Elegir la media de cada clase, en caso de que los atributos sean numéricos. Si no lo son, elegir el elemento más parecido, en promedio, con las otras instancias de su clase.

2. Calcular el radio de cada clase, midiendo la distancia de cada instancia de la clase con su media y consideramos la de mayor distancia.

3. Se crea la raíz del árbol con todas las clases.

4. Construcción de la dicotomía: (este proceso se realiza recursivamente hasta obtener una clase en cada hoja).

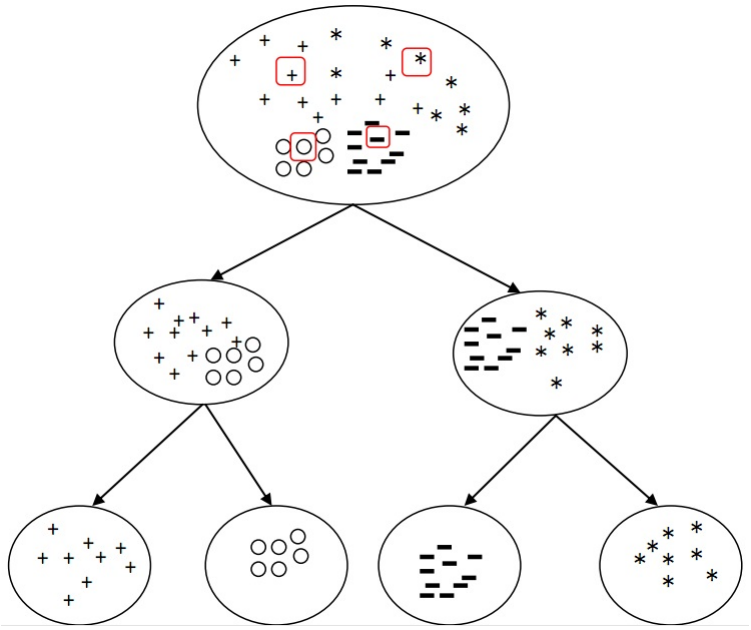

Figura 4.4. Método DAAR para construir dicotomías anidadas basadas en agrupamientos y radios.

a) Elegir las 2 clases cuya distancia D (equación 4.1) sea mayor, para formar los dos grupos

b) Cada clase, del nodo padre, se agrupará con el centro más cercano de cada grupo de acuerdo a la distancia D y en caso de empates, se toma el primer grupo.

c) Se crea un nodo hijo para cada uno de los grupos y se repite el proceso recursivamente.

5. Cuando el árbol se ha construido, se entrenan los clasificadores binarios correspondientes en cada nodo interno del árbol. Las clases se forman tomando las instancias del conjunto de entrenamiento que están contenidas en las clases agrupadas en cada nodo hijo.

La fase de clasificación es la misma que la del método DAA.

4.3. Método DAAP

Un inconveniente del método DAAR es que si un elemento de una clase se encuentra muy lejano de los demás, el radio de dicha clases sería muy grande, lo que pudiera indicar que hay mucho traslape, pero probablemente no sea cierto, ver figura 4.5. Considerando esto, se propuso una variante para calcular el radio considerando todas las instancias, es decir, calculando la distancia del centroide de la clase con todas las instancias de la misma y tomando como radio el promedio de estas distancias. De esta manera, la fórmula de la distancia D se modificó, quedan-

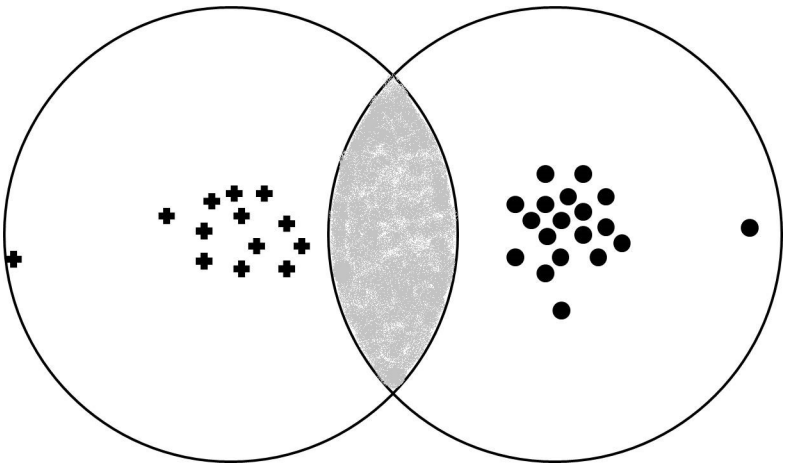

Figura 4.5. Ejemplo del problema de la obtención del radio de una clase como la distancia a la instancia más lejana.

do como se muestra en 4.2, donde r'_1 representa el promedio de las distancias del centroide de la clase C_1 (M_1) con las instancias de esta clase; y r'_2 es el promedio de las distancias del centroide de la clase C_2 (M_2) con todas las instancias de la clase C_2. A este método lo llamamos Dicotomía Anidada basada en Agrupamiento

y Promedio de radios DAAP. La construcción del clasificador DAAP es similar que en el método DAAR, solo se cambio la fórmula de la distancia D, utilizada para comparar medias (centroides) de las clases.

$$D(C_1, C_2) = \frac{d(M_1, M_2)}{r'_1 + r'_2} \tag{4.2}$$

Adicionalmente, en nuestro estudio proponemos construir ensambles de dicotomías anidadas usando los métodos DAA, DAAR y DAAP como base para los ensambles. Se realizaron ensambles de tipo *Bagging*, y los llamamos EDAA-b, EDAAR-b y EDAAP-b. Ensambles de tipo *AdaBoost* que llamamos EDAA-a EDAAR-a y EDAAP-a y ensambles tipo *MultiBoost* que nombramos EDAA-m, EDAAR-m y EDAAP-m.

4.4. Resultados experimentales

En esta sección se muestran los experimentos realizados con los métodos propuestos DAA, DAAR y DAAP, y los métodos base de comparación que construyen dicotomías anidadas de forma aleatoria ND de [Frank and Kramer, 2004], ND (*Class-Balanced*) y ND (*Data-Balanced*) de [Dong et al., 2005]. Además, se comparan los diferentes ensambles de los métodos propuestos contra los ensambles de dicotomías anidadas construidas por ND, ND (*Class-Balanced*) y ND (*Data-Balanced*).

Para los experimentos se emplearon 20 bases de datos del repositorio de aprendizaje automático UCI [Asuncion and Newman, 2007], las cuales se encuentran

especificadas en la tabla 4.1. Elegimos estas bases porque son las más comúnmente utilizadas para evaluar los métodos de construcción de dicotomías anidadas.
Todos los experimentos se realizaron utilizando validación cruzada de 10 pliegues, cuidando que las particiones de cada pliegue fueran las mismas para todos los métodos. Los métodos propuestos se programaron en Java con plataforma NetBeans 6.9, los métodos ND, ND (*Class Balanced*) , ND (*Data Balanced*) y END se tomaron de weka 3.7.1 [Holmes et al., 1994]. Todos los experimentos se hicieron en una computadora con procesador Pentium Dual-Core de 2.9 GHz, con 3 GB de RAM y con sistema operativo Linux-Ubuntu 11.04.
Como clasificadores binarios base, tanto para los métodos propuestos como para los métodos de comparación, se utilizaron C4.5, Random Forest, 1-NN, 3-NN y *Naive Bayes*. Todos en su implementación en weka 3.7.1

4.5. Experimentos con los método DAA, DAAR y DAAP

Para mostrar el desempeño de los métodos DAA, DAAR y DAAP, se compararon sus resultados con los resultados de precisión de clasificación de los métodos ND, ND(DB) que corresponde al método ND (*Data-Balanced*) y ND(CB) que corresponde al método ND (*Class-Balanced*). En la tabla 4.2 se muestran los promedios de precisión para las 20 bases de datos, obtenidos con la validación cruzada de 10 pliegues, cuando se utiliza cada uno de los clasificadores binarios, como clasificador base de las dicotomías anidadas. Colocando en cada columna el método

Tabla 4.1. Características de las bases de datos utilizadas en los experimentos.

Num	**Base de datos**	**Instancias**	**Numericos**	**Nominales**	**Clases**
1	Anneal	898	6	32	6
2	Audiology	226	0	69	24
3	Balance-scale	625	4	0	3
4	Car	1728	0	6	4
5	Dermatology	366	1	33	6
6	Mfeat-factors	2000	216	0	10
7	Mfeat-Karhunen	2000	64	0	10
8	Mfeat-morphological	2000	6	0	10
9	Mfeat-pixel	2000	0	240	10
10	Nursey	12960	0	8	5
11	Optdigits	5620	64	0	10
12	Page-blocks	5473	10	0	5
13	Pendigits	10992	16	0	10
14	Primary-tumor	339	0	17	22
15	Segment	2310	19	0	7
16	Soybean	683	0	35	19
17	Vehicle	846	18	0	4
18	Vowel-context	990	10	2	11
19	Waveform	5000	40	0	3
20	Zoo	101	1	15	7

evaluado y en cada fila el clasificador base utilizado. Además, en la última fila muestran los promedios generales de todos los clasificadores.

En la tabla 4.2 se puede observar que para el clasificador C4.5 el método DAA fue el que obtuvo mayor precisión en promedio, para Random Forest el método que obtuvo mayor precisión fue ND (DB), para 1-NN el método DAAR fue el que obtuvo mayor precisión, para 3-NN el método que obtuvo mayor precisión fue ND

Tabla 4.2. Resultados en promedio de la precisión obtenida por cada método, de acuerdo al clasificador base utilizado.

Clasif. Base	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
C4.5	**88.8195**	88.5840	88.4827	87.2257	87.6372	88.0206
RandomForest	93.0740	92.8928	92.7311	93.5164	**93.6392**	93.5278
1-NN	93.6562	**93.7450**	93.5072	93.6420	93.6379	93.6550
3-NN	88.9724	88.7688	88.7842	88.9241	88.8712	**88.9948**
Naive Bayes	77.3286	**78.3513**	78.0168	73.1082	72.5946	72.6048
Prom. Gral.	88.3701	**88.4684**	88.3044	87.2833	87.2760	87.3606

(CB) y para Naive Bayes el método DAAR fue el que obtuvo la mejor precisión. Los métodos propuestos obtienen los promedios generales más altos.

En la tabla 4.3 se muestran los resultados promedio de la precisión obtenida por cada método utilizando un ensamble tipo Bagging que utiliza las dicotomías anidadas como clasificador base. En la tabla 4.3 se observa que ND (CB) fue el que obtuvo mejores resultados de precisión para C4.5 y Random Forest. END(DB)-b fue el que mejor precisión promedio obtuvo para 1-NN, y el método EDAA-a fue el que mejores resultados de precisión obtuvo para 3-NN y Naive Bayes. Aunque en este experimento no todos los métodos propuestos obtienen los promedios generales más altos, con el método EDAA-a se obtuvo el promedio general más alto.

En la tabla 4.4 se muestran los resultados promedio de la precisión obtenida por cada método utilizando el ensamble tipo *AdaBoost*. En la tabla 4.4 se observa que el método END (CB)-a obtuvo las mejores precisiones en promedio para C4.5, RandomForest y 3-NN, el método EDAA-a obtuvo las mejores precisiones

Tabla 4.3. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo Bagging.

Clasif. Base	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
C4.5	91.6088	91.7295	91.7071	92.3378	92.2242	**92.6853**
RandomForest	94.3272	94.1790	93.9499	94.4792	94.3878	**94.5094**
1-NN	93.3340	93.3695	93.3201	93.4056	**93.4366**	93.4152
3-NN	**90.2490**	89.2979	89.3237	90.0167	90.0109	89.9953
Naive Bayes	**82.6152**	80.5675	79.4916	80.3264	80.4914	80.6551
Prom. Gral.	**90.4268**	89.8287	89.5585	90.1131	90.1102	90.2521

Tabla 4.4. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo AdaBoost.

Clasif. Base	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
C4.5	93.8316	93.6975	93.4738	93.7985	93.6440	**93.9763**
RandomForest	94.3617	94.2332	94.3696	94.5557	94.3758	**94.5869**
1-NN	**93.3168**	93.2863	93.2863	93.1964	93.2810	93.1986
3-NN	92.5182	92.4990	92.4625	92.5537	92.4805	**92.6593**
Naive Bayes	83.5573	**83.6234**	82.7055	80.2247	81.1566	81.3165
Prom. Gral.	**91.5171**	91.4679	91.2595	90.8658	90.9876	91.1475

en promedio para 1-NN y el método EDAAR-a obtuvo las mejores precisiones en promedio con Naive Bayes. Los métodos propuestos obtienen los promedios generales más altos.

En la tabla 4.5 se muestran los resultados obtenidos en promedio por cada método, utilizando un ensamble tipo MultiBoost que utiliza las dicotomías anidadas como clasificador base. En la tabla 4.5 se observa que el método END(DB)-m obtuvo los mejores resultados en promedio con C4.5 y Random Forest, el método EDAAR-m fue el que obtuvo mejores resultados de precisión con 1-NN, 3-NN y Nive Bayes. Los métodos propuestos obtienen los promedios generales más altos.

Tabla 4.5. Resultados en promedio de la precisión obtenida por cada método utilizando el ensamble tipo MultiBoost.

Clasif. Base	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
C4.5	93.3766	93.2940	93.2147	93.4070	**93.5987**	93.4758
RandomForest	94.3022	94.1228	94.1703	94.3481	**94.3668**	94.3200
1-NN	93.3557	**93.4742**	93.3338	93.3033	93.3140	93.2784
3-NN	91.1117	**91.3200**	91.3450	91.2266	91.2027	91.0203
Naive Bayes	82.3955	**83.4441**	82.3994	80.4584	80.7041	80.1636
Prom. Gral.	90.9084	**91.1310**	90.8926	90.5487	90.6373	90.4516

A partir de estos experimentos, se puede apreciar que los métodos propuestos obtienen en la mayoría de los casos los promedios generales más altos. En la tabla 4.3 aunque la mayoría de los promedios de END-b y END(DB)-b y END(CB)-b son más altos que los métodos propuestos EDAAR-b y EDAAP-b, el método propuesto EDAAP-b es el que obtiene el promedio general más alto.

Puesto que los promedios pueden en ocasiones ser engañosos, se realizaron prueba de Wilcoxon [Demšar, 2006] de significancia estadística con un 95 % de nivel de confianza, pero no se obtuvieron resultados favorables con los métodos propuestos, debido a que en la mayoría de los casos se obtuvieron empates. Lo que muestra que en promedio los métodos propuestos son competitivos, sin embargo al ser no aleatorios garantizan que siempre se encuentraran buenas dicotomías anidadas, lo cual no puede garantizarse con los otros métodos aleatorios.
Adicionalmente, realizamos un conteo de cuántas veces se gana, emparta o pierde cada método. En las siguientes tablas se muestra cuántas veces gana-empata-pierde el método de la fila comparado con cada método de las columnas. En la

tabla 4.6 se muestra cuántas veces gana, empata o pierde cada método de las filas con los métodos de las columnas utilizando C4.5 como clasificador base de las dicotomías. De la tabla 4.6 puede notarse que con C4.5 los métodos propuestos superan a los otros métodos comparados, en la mayoría de las bases de datos.

Tabla 4.6. Comparación método a método con el clasificador base C4.5.

Método	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
DAA	0	10-3-7	14-1-5	13-2-5	10-2-8	11-1-8
DAAR	7-3-10	0	10-1-9	12-1-7	9-1-10	9-0-11
DAAP	5-1-14	9-1-10	0	13-1-6	13-2-5	15-0-5
ND	5-2-13	7-1-12	6-1-13	0	8-2-10	8-0-12
ND(DB)	8-2-10	10-1-9	5-2-13	10-2-8	0	5-5-10
ND(CB)	8-1-11	11-0-9	5-0-15	12-0-8	10-5-5	0

En la tabla 4.7 se muestran los resultados comparando método a método con *Random Forest* como clasificador base. De la tabla 4.7 puede notarse que ND, ND(DB) y ND(CB) superan en la mayoría de las bases de datos a los métodos propuestos.

Tabla 4.7. Comparación método a método con el clasificador base *Random Forest*.

Método	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
DAA	0	14-1-5	15-2-3	7-1-12	8-1-11	9-1-10
DAAR	5-1-14	0	11-1-8	4-0-16	4-0-16	5-0-15
DAAP	3-2-15	8-1-11	0	3-3-14	2-2-16	5-1-14
ND	12-1-7	16-0-4	14-3-3	0	7-3-10	8-1-11
ND(DB)	11-1-8	16-0-4	16-2-2	10-3-7	0	12-3-5
ND(CB)	10-1-9	15-0-5	14-1-5	11-1-8	5-3-15	0

En la tabla 4.8 se muestran los resultados con 1-NN. De la tabla 4.8 puede notarse que los métodos propuestos superan a los demás métodos comparados, en la mayoría de las bases de datos.

Tabla 4.8. Comparación método a método con el clasificador base 1-NN.

Método	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
DAA	0	8-1-11	10-1-9	8-6-6	9-6-5	11-3-6
DAAR	11-1-8	0	12-6-2	14-1-5	13-3-4	11-4-5
DAAP	9-1-10	2-6-12	0	11-1-8	10-0-10	9-1-10
ND	6-6-8	5-1-14	8-1-11	0	10-5-5	10-6-4
ND(DB)	5-6-9	4-3-13	10-0-10	5-5-10	0	5-11-4
ND(CB)	6-3-11	5-4-11	9-1-10	4-6-10	4-11-5	0

En la tabla 4.9 se muestran los resultados de comparar método a método con 3-NN como clasificador base. En la tabla 4.9 se observa que los métodos propuestos superan a los demás métodos comparados, en la mayoría de las bases de datos.

Tabla 4.9. Comparación método a método con el clasificador base 3-NN

Método	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
DAA	0	14-0-6	11-1-8	9-5-6	11-4-5	12-2-6
DAAR	6-0-14	0	7-4-9	7-3-10	10-0-10	7-4-9
DAAP	8-1-11	9-4-7	0	9-1-10	10-2-8	9-2-9
ND	6-5-9	10-3-7	10-1-9	0	11-2-7	9-4-7
ND(DB)	5-4-11	10-0-10	8-2-10	7-2-11	0	7-4-9
ND(CB)	6-2-12	9-4-7	9-2-9	7-4-9	9-4-7	0

En la tabla 4.10 se muestran los resultados utilizando el clasificador *Naive Bayes*. En la tabla 4.10 puede notarse que los métodos propuestos superan a los demás métodos comparados, en la mayoría de las bases de datos.

Tabla 4.10. Comparación método a método con el clasificador base *Naive Bayes*.

Método	DAA	DAAR	DAAP	ND	ND(DB)	ND(CB)
DAA	0	7-2-11	11-0-9	13-0-7	13-0-7	12-0-8
DAAR	11-2-7	0	13-0-7	13-1-6	14-0-6	14-0-6
DAAP	9-0-11	7-0-13	0	13-1-6	14-0-6	15-0-5
ND	7-0-13	6-1-13	6-1-13	0	12-0-8	11-0-9
ND(DB)	7-0-13	6-0-14	6-0-14	8-0-12	0	11-2-7
ND(CB)	8-0-12	6-0-14	5-0-15	9-0-11	7-2-11	0

A partir de los resultados mostrados en las tablas 4.6, 4.7, 4.8, 4.9 y 4.10, se puede observar que con la mayoría de los clasificadores base utilizados, los métodos propuestos ganan en la mayoría de las bases de datos, a los otros métodos comparados; excepto cuando se usa como clasificador base *Random Forest* (ver tabla 4.7).

En las tablas 4.11 a 4.15 se muestra la comparación método a método mostrando cuántas veces gana-empata-pierde el método de la fila con cada método de la columna, en las 20 bases de datos evaluadas para los ensambles de dicotomías usando *Bagging*. En estas tablas puede notarse que con la mayoría de los clasificadores los métodos que más veces ganaron fueron END-b, END(DB)-b y END(CB)-b; mientras que, con el clasificador *Naive Bayes* como clasificador binario de las dicotomías anidadas, solo el método propuesto EDAA-b ganó más veces que los otros métodos comparados.

Tabla 4.11. Comparación método a método Bagging con C4.5.

Método	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
EDAA-b	0	6-0-14	7-0-13	4-1-15	3-0-17	1-0-19
EDAAR-b	14-0-6	0	10-0-10	3-0-17	4-1-15	3-1-16
EDAAP-b	13-0-7	10-0-10	0	2-0-18	4-1-15	2-1-17
END-b	15-1-4	17-0-3	18-0-2	0	13-0-7	10-0-10
END(DB)-b	17-0-3	15-1-4	15-1-4	7-0-13	0	5-4-11
END(CB)-b	19-0-1	16-1-3	17-1-2	10-0-10	11-4-5	0

Tabla 4.12. Comparación método a método Bagging con Random Forest

Método	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
EDAA-b	0	14-0-6	13-2-5	7-1-12	9-2-9	8-1-11
EDAAR-b	6-0-14	0	13-2-5	3-1-16	6-0-14	4-0-16
EDAAP-b	5-2-13	5-2-13	0	3-1-16	3-2-15	2-1-17
END-b	12-1-7	16-1-3	16-1-3	0	10-2-8	10-1-9
END(DB)-b	9-2-9	14-0-6	15-2-13	8-2-10	0	2-7-11
END(CB)-b	11-1-8	16-0-4	17-1-2	9-1-10	11-7-2	0

Tabla 4.13. Comparación método a método Bagging con 1-NN

Método	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
EDAA-b	0	6-2-12	9-2-9	6-2-12	4-1-15	5-0-15
EDAAR-b	12-2-6	0	12-4-4	10-1-9	8-3-9	8-3-9
EDAAP-b	9-2-9	4-4-12	0	7-2-11	5-3-12	6-2-12
END-b	12-2-6	9-1-10	11-2-7	0	2-9-9	3-9-8
END(DB)-b	15-1-4	9-3-8	12-3-5	9-9-2	0	6-12-2
END(CB)-b	15-0-5	9-3-8	12-6-6	8-9-3	2-12-6	0

Tabla 4.14. Comparación método a método Bagging con 3-NN

Método	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
EDAA-b	0	19-1-0	18-1-1	13-2-5	11-1-8	11-1-8
EDAAR-b	0-1-19	0	8-3-9	2-1-17	2-2-16	2-1-17
EDAAP-b	1-1-18	9-3-8	0	2-1-17	3-1-16	3-1-16
END-b	5-2-13	17-1-2	17-1-2	0	5-7-8	7-6-7
END(DB)-b	8-1-11	16-2-2	16-1-3	8-7-5	0	6-8-8
END(CB)-b	8-1-11	17-1-2	16-1-3	7-6-7	8-8-6	0

Tabla 4.15. Comparación método a método Bagging con Naive Bayes

Método	EDAA-b	EDAAR-b	EDAAP-b	END-b	END(DB)-b	END(CB)-b
EDAA-b	0	16-0-4	19-0-1	14-1-5	15-1-4	13-2-5
EDAAR-b	4-0-16	0	13-0-7	11-0-9	9-2-9	9-0-11
EDAAP-b	1-0-19	7-0-13	0	8-0-12	9-0-11	7-0-13
END-b	5-1-14	9-0-11	12-0-8	0	8-1-11	10-1-9
END(DB)-b	4-1-15	9-2-9	11-0-9	11-1-8	0	8-3-9
END(CB)-b	5-2-13	11-0-9	13-0-7	9-1-10	9-3-8	0

En las tablas 4.16 a 4.20 se muestra la comparación método a método mostrando cuántas veces gana-empata-pierde el método de la fila con el método de la columna, en las 20 bases de datos evaluadas para los ensambles de dicotomías usando *AdaBoost*. En estas tablas puede notarse que los métodos propuestos son mejores cuando se usa 1-NN, 3-NN y Naive Bayes, mientras que END-m, END(DB)-m y END(CB)-m son mejores cuando se usa C4.5 y Random Forest como clasificador binario de las dicotomías anidadas.

Tabla 4.17. Comparación método a método AdaBoost con Random Forest

Método	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
EDAA-a	0	13-1-6	12-1-7	6-2-12	11-2-7	8-1-11
EDAAR-a	6-1-13	0	8-1-11	4-1-15	6-1-13	4-1-15
EDAAP-a	7-1-12	11-1-8	0	5-3-12	8-2-10	8-1-11
END-a	12-2-6	15-1-4	12-3-5	0	9-4-7	11-1-8
END(DB)-a	7-2-11	13-1-6	10-2-8	7-4-9	0	7-2-11
END(CB)-a	11-1-8	15-1-4	11-1-8	8-1-11	11-2-7	0

Tabla 4.18. Comparación método a método AdaBoost con 1NN

Método	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
EDAA-a	0	8-1-11	9-2-9	10-2-8	9-3-8	9-0-11
EDAAR-a	11-1-8	0	8-3-9	9-3-8	9-4-7	10-3-7
EDAAP-a	9-2-9	9-3-8	0	8-3-9	12-1-7	10-3-7
END-a	8-2-10	8-3-9	9-3-8	0	9-3-8	9-2-9
END(DB-a)	8-3-9	7-4-9	7-1-12	8-3-9	0	6-4-10
END(CB)-a	11-0-9	7-3-10	7-3-10	9-2-9	10-4-6	0

Tabla 4.16. Comparación método a método AdaBoost con C4.5

Método	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
EDAA-a	0	10-2-8	11-2-7	10-0-10	11-2-7	9-1-10
EDAAR-a	8-2-10	0	11-0-9	8-0-12	9-1-10	10-2-8
EDAAP-a	7-2-11	9-0-11	0	5-4-11	9-1-10	7-1-12
END-a	10-0-10	12-0-8	11-4-5	0	11-1-8	11-2-7
END(DB)-a	7-2-11	10-1-9	10-1-9	8-1-11	0	8-2-10
END(CB)-a	10-1-9	8-2-10	12-1-7	7-2-11	10-2-8	0

Tabla 4.19. Comparación método a método AdaBoost con 3NN

Método	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
EDAA-a	0	9-3-8	9-3-8	12-1-7	11-2-7	9-2-9
EDAAR-a	8-3-9	0	8-0-12	7-3-10	9-2-9	7-5-8
EDAAP-a	8-3-9	12-0-8	0	12-1-7	11-1-8	10-2-8
END-a	7-1-12	10-3-7	7-1-12	0	11-1-8	9-4-7
END(DB)-a	7-2-11	9-2-9	8-1-11	8-1-11	0	6-4-10
END(CB)-a	9-2-9	8-5-7	8-2-10	7-4-9	10-4-6	0

Tabla 4.20. Comparación método a método AdaBoost con Naive Bayes

Método	EDAA-a	EDAAR-a	EDAAP-a	END-a	END(DB)-a	END(CB)-a
EDAA-a	0	12-1-7	14-0-6	15-0-5	13-0-7	13-1-6
EDAAR-a	7-1-12	0	14-1-5	14-0-6	11-2-7	12-2-6
EDAAP-a	6-0-14	5-1-14	0	9-1-10	9-0-11	8-1-11
END-a	5-0-15	6-0-14	10-1-9	0	7-0-13	9-1-10
END(DB)-a	7-0-13	7-2-11	11-0-9	13-0-7	0	11-2-7
END(CB)-a	6-1-13	6-2-12	11-1-8	10-1-9	7-2-11	0

En las tablas 4.21 a 4.25 se muestra la comparación método a método mostrando cuántas veces gana-empata-pierde el método de la fila con el método de la columna, en las 20 bases de datos evaluadas para los ensambles de dicotomías usando Multiboost. En estas tablas puede notarse que los métodos propuestos son mejores cuando se usa 1-NN, 3-NN y Naive Bayes mientras que END-m, END(DB)-m y END(CB)-m son mejores cuando se usa C4.5 y Random Forest.

Tabla 4.22. Comparación método a método Multiboost con Random Forest

Método	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
EDAA-m	0	13-0-7	10-4-6	9-2-9	6-2-12	6-3-11
EDAAR-m	7-0-13	0	8-1-11	7-0-13	4-0-16	4-1-15
EDAAP-m	6-4-10	11-1-8	0	8-2-10	4-2-14	7-0-13
END-m	9-2-9	13-0-7	10-2-8	0	10-1-9	8-2-10
END(DB)-m	12-2-6	16-0-4	14-2-4	9-1-10	0	6-5-9
END(CB)-m	11-3-6	15-1-4	13-0-7	10-2-8	9-5-6	0

Tabla 4.23. Comparación método a método Multiboost con 1-NN

Método	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
EDAA-m	0	7-5-8	8-7-5	10-3-7	10-3-7	9-5-6
EDAAR-m	8-5-7	0	12-3-5	13-2-5	11-4-5	15-2-3
EDAAP-m	5-7-8	5-3-12	0	10-3-7	9-3-8	9-5-6
END-m	7-3-10	5-2-13	7-3-10	0	6-4-10	8-4-8
END(DB)-m	7-3-10	5-4-11	8-3-9	10-4-6	0	7-8-5
END(CB)-m	6-5-9	3-2-15	6-5-9	8-4-8	5-8-7	0

Tabla 4.21. Comparación método a método Multiboost con C4.5

Método	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
EDAA-m	0	10-2-8	12-1-7	5-3-12	7-2-11	9-1-10
EDAAR-m	8-2-10	0	10-0-10	6-1-13	6-3-11	7-3-10
EDAAP-m	7-1-12	10-0-10	0	7-2-11	6-1-13	7-1-12
END-m	12-3-5	13-1-6	11-2-7	0	8-3-9	8-2-10
END(DB)-m	11-2-7	11-3-6	13-1-6	9-3-8	0	10-3-7
END(CB)-m	10-1-9	10-3-7	12-1-7	10-2-8	7-3-10	0

Tabla 4.24. Comparación método a método Multiboost con 3-NN

Método	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
EDAA-m	0	7-3-10	4-2-14	9-2-9	8-0-12	11-3-6
EDAAR-m	10-3-7	0	7-5-8	11-2-7	12-0-8	13-0-7
EDAAP-m	14-2-4	8-5-7	0	8-4-8	10-2-8	11-1-6
END-m	9-2-9	7-2-11	8-4-8	0	10-2-8	10-2-8
END(DB)-m	12-0-8	8-0-12	8–2-10	8-2-10	0	10-2-8
END(CB)-m	6-3-11	7-0-13	6-1-11	8-2-10	8-2-10	0

Tabla 4.25. Comparación método a método Multiboost con Naive Bayes

Método	EDAA-m	EDAAR-m	EDAAP-m	END-m	END(DB)-m	END(CB)-m
EDAA-m	0	6-1-13	8-1-11	8-3-9	12-2-6	12-1-7
EDAAR-m	13-1-6	0	14-1-5	14-0-6	14-0-6	16-0-4
EDAAP-m	11-1-8	5-1-14	0	11-0-9	11-0-9	11-0-9
END-m	9-3-8	6-0-14	9-0-11	0	13-1-6	12-1-7
END(DB)-m	6-2-12	6-0-14	9-0-11	6-1-13	0	8-3-9
END(CB)-m	7-1-12	4-0-16	9-0-11	7-1-12	9-3-8	0

Con estos experimentos puede notarse que para los ensambles con Bagging los métodos END-b, END(DB)-b y END(CB)-b son mejores que los métodos propuestos. Con AdaBoost y MultiBoost los métodos propuestos son mejores cuando se usa 1-NN, 3-NN y Naive Bayes como clasificadores binarios de las dicotomías anidadas, mientras que END-a, END-m, END(DB)-a, END(DB)-m, END(CB)-a y END(CB)-m son mejores cuando se usa C4.5 y Random Forest.

Capítulo 5

Conclusiones

En este capítulo se muestran el sumario, las conclusiones y aportaciones obtenidas en esta tesis, así como posibles líneas para trabajo futuro.

5.1. Sumario

En esta tesis se propusieron tres métodos para construir dicotomías anidadas DAA, DAAR y DAAP. Estos métodos propuestos a diferencia de los métodos existentes no construyen la dicotomía anidada de manera aleatoria.
La idea de estos tres métodos es separar en los niveles superiores las clases más fáciles de separar y en los niveles inferiores las más difíciles. El primer método determina cuáles son las clases más fáciles de separar midiendo la distancia entre sus medias, considerando que las clases con medias más lejanas son más fáciles de separar. En el segundo método propuesto, además de la distancia entre las medias se considera también el radio de las clases, entendiéndose como radio la distancia de la media de la clase a la instancia más lejana dentro de la misma clase. En el

tercer método se sigue la idea de considerar los radios, pero como radio de una clase se considera el promedio de las distancias entre la media de la clase y el resto de las instancias de la misma.
Además, con los métodos propuestos se realizaron 3 tipos de ensamble uno de tipo Bagging, otro de tipo AdaBoost y otro de tipo Multiboost.

5.2. Conclusiones

Los métodos propuestos DAA, DAAR y DAAP obtuvieron mejores resultados en promedio que los otros métodos ND, ND (*Class-Balanced*) y ND (Data-Balanced). El método propuesto DAA obtuvo los mejores resultados de clasificación cuando se utiliza C4.5 como clasificador binario, la mejoría de los resultados de precisión se reflejan principalmente en las bases de datos con un mayor número de clases. El método DAAR también obtuvo los mejores resultados de precisión con los clasificadores 1-NN y Naive Bayes como clasificadores binarios. Con los clasificadores Random Forest y 3-NN los métodos propuestos obtienen en promedio precisiones competitivas con las precisiones alcanzadas por los métodos aleatorios de construcción de dicotomías anidadas.
En el ensamble tipo Bagging se mostró que el método propuesto DAA obtuvo, en promedio, mejores resultados de clasificación con los clasificadores 3-NN y Naive Bayes. A pesar de que el método END(CB)-b obtuvo los mejores resultados de clasificación con los clasificadores C4.5 y Random Forest, el método DAA obtuvo el mayor promedio de forma general.
Para los ensambles de tipo AdaBoost los métodos propuestos obtuvieron los mejores resultados de clasificación en la mayoría de los casos. El método DAA obtuvo

los mejores resultados con el clasificador binario 1-NN y el método DAAR fue mejor con Naive Bayes. Mientras que el método END(CB)-a fue mejor con los clasificadores binarios C4.5, Random Forest y 3-NN.
Con Multiboost se mostró que los métodos propuestos obtuvieron los mejores resultados promedios de clasificación, destacándose principalmente el método DAAR con los clasificadores binarios 1-NN, 3NN y Naive Bayes. Mientras que END(DB)-m fue mejor utilizando como clasificadores binarios a C4.5 y Random Forest.

5.3. Aportación

Tres nuevos métodos no aleatorios para construir dicotomías anidadas de buena calidad, que ayudan a resolver problemas multiclase. Los métodos propuestos son competitivos con los métodos aleatorios. Sin embargo, al no ser aleatorios garantizan que siempre se encuentran buenas dicotomías anidadas, lo cual no puede garantizarse con los métodos aleatorios.

5.4. Trabajo Futuro

Se propone considerar otras formas de medir el traslape entre las clases, esto es, medir cuántos objetos hay en regiones donde la mayoría de los objetos son de otras clases, lo que provoca ruido en los datos.
Además para mejorar la calidad de clasificación se piensa aplicar métodos de selección de atributos y/o instancias en los clasificadores binarios.

Bibliografía

[Ahn et al., 2007] Ahn, H., Moon, H., Fazzari, M. J., Lim, N., Chen, J. J., and Kodell, R. L. (2007). Classification by ensembles from random partitions of high-dimensional data. *Comput. Stat. Data Anal.*, 51:6166–6179.

[Allwein et al., 2001] Allwein, E. L., Schapire, R. E., and Singer, Y. (2001). Reducing multiclass to binary: a unifying approach for margin classifiers. *J. Mach. Learn. Res.*, 1:113–141.

[Aoki and Kudo, 2002] Aoki, K. and Kudo, M. (2002). Decision tree using class-dependent feature subsets. pages 761–769.

[Aoki and Kudo, 2010] Aoki, K. and Kudo, M. (2010). A top-down construction of class decision trees with selected features and classifiers. pages 390–398.

[Asuncion and Newman, 2007] Asuncion, A. and Newman, D. (2007). Uci machine learning repository.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Mach. Learn.*, 24:123–140.

[Debnath et al., 2004] Debnath, R., Takahide, N., and Takahashi, H. (2004). A decision based one-against-one method for multi-class support vector machine. *Pattern Anal. Appl.*, 7:164–175.

[Demšar, 2006] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

[Dong et al., 2005] Dong, L., Frank, E., and Kramer, S. (2005). Ensembles of balanced nested dichotomies for multi-class problems. pages 84–95.

[Frank and Kramer, 2004] Frank, E. and Kramer, S. (2004). Ensembles of nested dichotomies for multi-class problems. pages 305–312.

[Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting.

[Fürnkranz, 2002] Fürnkranz, J. (2002). Round robin classification. *J. Mach. Learn. Res.*, 2:721–747.

[Garcia-Pedrajas and Ortiz-Boyer, 2006] Garcia-Pedrajas, N. and Ortiz-Boyer, D. (2006). Improving multiclass pattern recognition by the combination of two strategies. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28:1001–1006.

[Hastie and Tibshirani, 1998] Hastie, T. and Tibshirani, R. (1998). Classification by pairwise coupling. pages 507–513.

[Ho, 1998] Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20:832–844.

[Holmes et al., 1994] Holmes, G., Donkin, A., and Witten, I. H. (1994). Weka: a machine learning workbench.

[Huhn and Hüllermeier, 2008] Huhn, J. C. and Hüllermeier, E. (2008). Is an ordinal class structure useful in classifier learning? *IJDMMM*, 1(1):45–67.

[Quost et al., 2007a] Quost, B., Denux, T., and Masson, M.-H. (2007a). Pairwise classifier combination using belief functions. *Pattern Recogn. Lett.*, 28:644–653.

[Quost et al., 2007b] Quost, B., Denux, T., and Masson, M.-H. (2007b). Pairwise classifier combination using belief functions. *Pattern Recogn. Lett.*, 28:644–653.

[Rifkin and Klautau, 2004] Rifkin, R. and Klautau, A. (2004). In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141.

[Rodríguez et al., 2010] Rodríguez, J. J., García-Osorio, C., and Maudes, J. (2010). Forests of nested dichotomies. *Pattern Recogn. Lett.*, 31:125–132.

[Statistics and Breiman, 2001] Statistics, L. B. and Breiman, L. (2001). Random forests. pages 5–32.

[Webb, 2000] Webb, G. I. (2000). Multiboosting: A technique for combining boosting and wagging. pages 159–196.