RNPS No. 2142
ISSN 2072-6287
Versión Digital

SERIE AZUL

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

REPORTE TÉCNICO
Reconocimiento de Patrones

Combinación de clasificadores supervisados

Eric Javier Hernández Saura
y José Ruiz Shulcloper

RT_064 **octubre 2014**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

RNPS No. 2142
ISSN 2072-6287
Versión Digital

SERIE AZUL

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

REPORTE TÉCNICO

Reconocimiento de Patrones

Combinación de clasificadores supervisados

Eric Javier Hernández Saura
y José Ruiz Shulcloper

RT_064 **octubre 2014**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

Tabla de contenido

Combinación de clasificadores supervisados

Eric Javier Hernández Saura y José Ruiz Shulcloper

Equipo de Reconocimiento de Patrones, Centro de Aplicaciones de Tecnologías de Avanzada (CENATAV), La Habana, Cuba
{esaura, jshulcloper}@cenatav.co.cu

RT_064, Serie Azul, CENATAV
Aceptado: 23 de septiembre de 2014

Resumen. Existen problemas en el área del Reconocimiento de Patrones, específicamente en la clasificación supervisada, en los cuales la utilización de un solo clasificador no da resultados satisfactorios. En muchos de estos casos, el problema ha sido resuelto a través de la utilización de varios clasificadores y después combinando los resultados de estos. La combinación de clasificadores es una práctica aceptada por la comunidad de científicos en el área de Aprendizaje de Máquinas y del Reconocimiento de Patrones y uno de los campos en los que se continúa investigando en la actualidad. En este trabajo se hace un análisis crítico del estado del arte acerca de las principales metodologías utilizadas en la combinación de clasificadores, así como de las herramientas que han sido desarrolladas para su estudio.

Palabras clave: combinación de clasificadores, sistemas de múltiples clasificadores, esquema de clasificación.

Abstract. There are problems in the area of Pattern Recognition, specifically in the supervised classification, in which the use of a single classifier does not give satisfactory results. In many cases, the problem has been resolved through the use of various classifiers and then combining their results. The combination of classifiers is an accepted practice in the scientific community of Machine Learning and Pattern Recognition and also one of the fields with constant research activity. This paper provides a critical analysis of the state of the art about the main methods used in combining classifiers, as well as the tools that have been developed for its study.

Keywords: combination of classifiers, multiple classifier systems, ensemble of classifiers.

1 Introducción

En los campos de reconocimiento de patrones y aprendizaje de máquinas la clasificación supervisada es una de las tareas más demandadas. Se está en presencia de un problema de clasificación supervisada cuando se necesita saber a qué clase pertenece un objeto nunca antes visto, teniendo en cuenta cierta experiencia previa. Por ejemplo, si se cuenta con un conjunto de correos electrónicos divididos en dos grupos: aquellos correos que son basura, y aquellos que no; y se desea, a partir de este conjunto que representa la experiencia previa, crear o utilizar un modelo que sea capaz de discriminar nuevos correos no presentes en la muestra anterior en basura o no, entonces se está en presencia de un problema de clasificación supervisada.

En los últimos años numerosos algoritmos han surgido para resolver este tipo de tarea, todos tienen propiedades diferentes, siguen ideas diferentes, han sido pensados para resolver problemas diferentes;

por lo que en la actualidad se cuenta con una gran caja de herramientas, las cuales deben ser utilizadas según corresponda para el problema que se desea resolver. Sin embargo, estas herramientas no son perfectas ya que pueden cometer errores, por ejemplo considerar a un correo importante como basura. Lo que sucede en realidad es que estas herramientas son incapaces de realizar un trabajo perfecto debido al contexto en el que son utilizadas.

A los humanos la historia les ha demostrado que tener en cuenta varias opiniones es mejor que tener una sola. Esto sucede porque que se equivoque una sola persona tomando una decisión es más probable a que se equivoquen 10 personas tomando la misma decisión, asumiendo que estas personas tienen cierta capacidad que les permite enfrentarse al problema, por ejemplo un jurado. Además las diferentes opiniones pueden estar sustentadas en distintas bases de conocimiento lo que permite que se abarque el problema desde distintas aristas, por ejemplo cuando se va a tomar una decisión en medicina se reúnen el anestesiólogo, el ortopédico, el cirujano, entre otros.

Algo similar ha ocurrido con la clasificación supervisada y en el campo del aprendizaje de máquinas en general: se ha intentado resolver problemas para los cuales todavía no se tiene un método específico que lo resuelva de manera satisfactoria, a través de la utilización de varios métodos y combinando el resultado de estos. En el caso particular de la clasificación supervisada dos enfoques posibles serían:

- Resolver un problema dado a partir de combinar –integrar, fusionar– distintas soluciones previas al mismo problema.
- Dividir el problema en varios sub-problemas, resolver dichos sub-problemas con las técnicas tradicionales y después a partir de dichas soluciones arribar a la solución del problema más general.

La idea de construir herramientas que combinen otras herramientas ya existentes ha recibido mucha atención en los últimos años, sobre todo desde la segunda mitad de la década del 90 hasta la actualidad. Sin embargo, esta idea ya existía y era utilizada posiblemente desde los años 60. Muchos autores tienen opiniones encontradas en este aspecto, por ejemplo, en [1] se menciona el año 1962 con el trabajo [2] como el primero en seguir esta idea y en [3] se menciona el año 1977 con el trabajo [4].

Este es un trabajo acerca de cómo han sido utilizadas las metodologías antes mencionadas en el contexto de la clasificación supervisada. Intuitivamente un clasificador es un procedimiento –algoritmo, función– que ha sido creado a partir de la experiencia previa, un clasificador toma como parámetro un objeto que pudiera representar un correo o un paciente y lo clasifica, esto es, dice si el correo es basura o no, o la enfermedad que tiene el paciente. Entonces, más específicamente, en este trabajo se estudiarán las metodologías –las técnicas– que se han venido utilizando por la comunidad científica para combinar los resultados de los clasificadores. A continuación se dará una idea general acerca de cómo se expondrá este conocimiento al lector.

Primeramente es necesario definir un lenguaje común de entendimiento, dar definiciones formales acerca de lo que es un clasificador, un problema de clasificación supervisada, cuáles son los factores que entran en juego en dichos problemas, cómo se sabe si un clasificador resuelve un problema de manera correcta o no –o al menos de manera aceptable–, qué es una combinación de clasificadores. Este contenido será expuesto en el segundo epígrafe con nombre Conceptos generales.

En el tercer epígrafe, Funciones de combinación de las salidas de los clasificadores, se mostrarán al lector las distintas técnicas utilizadas en la combinación de las salidas de los clasificadores. Es decir, cómo se pueden utilizar los resultados obtenidos por varios métodos de forma independiente para formar un nuevo resultado y las propiedades de este.

Para combinar las salidas de los clasificadores, primeramente hay que tener los clasificadores. Ahora, ¿cuáles clasificadores nos conviene combinar?, ¿cómo los obtenemos? En el cuarto epígrafe, Obtención de los clasificadores individuales, se mostrarán algunas metodologías utilizadas para a partir de la experiencia previa obtener los distintos clasificadores que van a ser combinados.

Evidentemente combinar clasificadores que siempre den los mismos resultados no tiene sentido, es necesario que cuando algunos se equivoquen otros den la respuesta correcta. Esto se conoce como *diversidad* en el campo de la combinación de clasificadores y será estudiado en el epígrafe cinco.

En el sexto epígrafe, que se nombra Categorizaciones, se espera que el lector se encuentre familiarizado con las técnicas más utilizadas por la comunidad así como con la fundamentación de las mismas. Entonces, en dicho epígrafe se pasará a describir, cómo se han organizado los estudios en esta rama de la ciencia, es decir, se mostrarán las distintas taxonomías que han sido propuestas con el objetivo de organizar los algoritmos y teorías existentes en el campo.

En los epígrafes que siguen al sexto se darán las conclusiones y se expondrán distintas direcciones para futuras investigaciones. Finalmente se mostrarán las referencias de este trabajo.

2 Conceptos generales

En este epígrafe se darán las definiciones formales que serán utilizadas a través de todo el trabajo. Se presentará de manera formal qué es un problema de clasificación supervisada, y se definirán los conceptos necesarios para el estudio de las posibles soluciones que dichos problemas pueden tener utilizando los métodos de combinación de clasificadores. Entre estas definiciones está la de clasificador así como del procedimiento a través del cual este se obtiene. Es importante que se entienda que en este epígrafe se definirá un número mínimo de conceptos necesarios para el entendimiento del trabajo completo, para un estudio exhaustivo de estas cuestiones el lector puede remitirse a la literatura especializada en estos temas.

2.1 Clasificación supervisada

Sea un conjunto de objetos U llamado *universo*, cada objeto perteneciente a U puede ser descrito en términos de un conjunto de *rasgos.* A estos rasgos también se les llaman atributos, propiedades. De esta forma todo objeto de U se representa como un n-uplo $x = (r_1, r_2, \dots, r_n)$ donde r_i representa el valor tomado por el i-ésimo atributo para el objeto en cuestión. Será denotado por S el conjunto que define el *espacio de representación* de U, es decir, el espacio en el que los objetos de U se representan según los rasgos en cuestión. Es importante notar que los atributos pueden tomar valores en conjuntos distintos, convirtiendo a S para el caso más general en el siguiente producto cartesiano $(R_1 \times R_2 \times \dots \times R_n)$, donde R_i no es más que el conjunto de los valores posibles que puede tomar el atributo i-ésimo.

Se conoce –o asume– además que existen k subconjuntos propios de U, los cuales denotaremos por K_i, que cumplen que $\bigcup_{i=1}^{k} K_i = U$. O sea, los subconjuntos propios representan un cubrimiento de U y los denominaremos *clases*. Denotaremos por Ω al conjunto que contiene todos los subconjuntos K_i, k representa la cantidad de clases, o sea $k = |\Omega|$. A cada elemento de U se le puede asociar una k-tupla de pertenencia que codifique la relación de pertenencia del elemento con cada uno de los k subconjuntos K_i, sin embargo la relación de pertenencia de todos los elementos de U con cada uno de los subconjuntos no tiene por qué ser conocida. El espacio de k-tuplas de pertenencia será denotado por β^k y la k-tupla del objeto representado por x será denotada por $\beta(x)$.

La pertenencia de un elemento a un conjunto puede ser expresada tanto en términos de la teoría clásica de conjuntos –pertenece o no pertenece–, de la teoría de conjuntos difusos –expresando un grado de pertenencia– o cualquier otra teoría de conjuntos. Sin embargo en este trabajo, se utilizará solamente la teoría clásica de conjuntos para representar el cubrimiento subyacente al problema, esto es, los objetos pertenecen o no a cada uno de los subconjuntos K_i.

Cuando se tiene un universo de objetos U y una muestra de elementos por cada subconjunto K_i; y se desea generalizar la información aportada por estas muestras para saber a qué subconjunto(s) pertenece un nuevo objeto no presente en ellas, se está en presencia de un problema de *clasificación supervisada*. A dichas muestras se les llama *conjunto de entrenamiento* y será denotado por Z.

Un algoritmo que a partir de un conjunto de entrenamiento y posiblemente un conjunto adicional de parámetros, es capaz de decir a qué subconjuntos pertenece un objeto no presente en dichas muestras será denominado *algoritmo de clasificación*, en la literatura en inglés también se le conoce como *inducer*. Cada uno de estos algoritmos se caracteriza por tener un *criterio de clasificación*, una forma de procesar los datos que lo distingue. Un algoritmo de clasificación con un conjunto de entrenamiento y argumentos adicionales fijos define un *clasificador*. Un clasificador puede ser estudiado como una función $g: S \to T^k$, donde S es el espacio de representación de U y T^k es un espacio de tuplas de tamaño k con componentes que toman valores en algún conjunto T dado, de forma tal que el valor de la i-ésima componente codifique el grado de soporte que da el clasificador a la hipótesis de que el elemento a clasificar pertenezca al subconjunto K_i. Si se desea que el clasificador tenga la oportunidad de abstenerse la imagen de la función g debe ser $(T \cup \{*\})^k$ donde el símbolo $*$ codifica la abstención del clasificador.

En este trabajo los algoritmos de clasificación serán denotados por A, dicho algoritmo cuando recibe una muestra de entrenamiento Z produce un clasificador D, o sea $A(Z) = D$. Entonces $D(x)$ representa el resultado obtenido al clasificar el objeto representado por x a través del clasificador D. En ocasiones se hará uso de la notación $A(Z, x)$ para denotar el resultado obtenido al hacer $D(x)$, cuando $D = A(Z)$, otra forma de verlo es como el resultado obtenido del algoritmo de clasificación A con la muestra Z en el objeto representado por x.

Se dirá que los clasificadores que provienen de un mismo algoritmo de clasificación pero con distintos conjuntos de entrenamiento, o distintos parámetros adicionales forman una *familia de clasificadores*. En otras palabras, un algoritmo de clasificación es –o representa– una familia de clasificadores supervisados. Por ejemplo, los clasificadores que utilizan el algoritmo del vecino más cercano forman una familia, estos se diferencian en el conjunto de entrenamiento utilizado o en la función de distancia utilizada.

En el año 1992 Xu y Suen [5] proponen clasificar las salidas de los clasificadores según el nivel de información que estas aportan. En este sentido se proponen tres categorías:

1- Tipo abstracto. El clasificador solo dice si el objeto pertenece o no a cada uno de los subconjuntos K_i. En este caso $T = \{0,1\}$, es decir, las tuplas de salida del clasificador tienen solo valores de ceros y unos donde el uno (cero) en la posición número i codifica que el objeto pertenece (no pertenece) a K_i.
2- Tipo rango. Se establece un ordenamiento sobre los K_i que expresa el soporte del clasificador a la hipótesis de la pertenencia o no de un objeto a cada uno de estos. En este caso T es un subconjunto del conjunto de números naturales. El número natural que se encuentra en la posición i representa el lugar de K_i en el ordenamiento realizado. En realidad T no tiene que ser un subconjunto de los números naturales, pudiera ser cualquier conjunto totalmente ordenado.
3- Tipo medida. El clasificador da un grado de soporte a las hipótesis de que el objeto pertenezca a cada uno de los K_i. En este caso T es igual al conjunto de los números reales, para el caso más general, sin embargo también pudiera ser el intervalo $[0,1]$.

Nótese que las categorías anteriores no son necesariamente excluyentes, sino que representan más bien una jerarquía, ya que por ejemplo una salida de tipo medida se puede convertir en tipo rango y abstracta e igualmente una de tipo rango se puede convertir en tipo abstracto, aunque se pierda información en el proceso, es decir, este pudiera ser visto como un proceso de abstracción. Sin embargo es imposible convertir una salida tipo abstracto en tipo rango o medida, pues no se cuenta con la información necesaria para ello. Por ejemplo, la siguiente salida del tipo medida $(0.2, 0.5, 0.3)$ puede ser convertida en la siguiente salida de tipo rango $(3, 1, 2)$ pues $0.5 > 0.3 > 0.2$, y en la siguiente salida de tipo abstracto $(0, 1, 0)$ pues el soporte 0.5 es el mayor de todos.

2.2 Eficacia

Un clasificador puede cometer errores cuando clasifica un objeto, de hecho es importante que el clasificador cometa la menor cantidad de errores posibles, es decir, se necesita que el clasificador sea eficaz. Sea $g: S \to T^k$ un clasificador; y V una muestra de elementos de U, la cual se llamará *conjunto de validación*, donde a cada uno de los objetos presentes en la muestra se le conoce su tupla de pertenencia, y una función $\xi: (\beta^k \times T^k) \to \mathbb{R}$ –recuérdese que β^k representa el conjunto de tuplas de pertenencia, T^k representa el espacio de salida del clasificador g, y $\mathbb{R}$ el conjunto de los números reales– tal que para todo objeto $x \in V$ el valor $\xi(\beta(x), g(x))$ representa el costo de la pérdida –o la ganancia– asociada al hecho de que el clasificador g dé como resultado $g(x)$ cuando la tupla de pertenencia de x es $\beta(x)$. Existen diversas formas de medir la eficacia del clasificador g, a partir del conjunto V y la función ξ.

Por ejemplo, considérese que los K_i forman una partición sobre U y que g da salidas de tipo abstracto, podemos considerar a ξ como una función que devuelve 0 si la tupla de pertenencia y la salida de g con respecto a un objeto coinciden y 1 en cualquier otro caso, es decir, se dice que los errores tienen todos un mismo costo igual a 1 y que las clasificaciones correctas no tienen costo. Entonces es posible calcular la eficacia de g en V de la siguiente forma:

$$eficacia(g, V) = 1 - \left(\frac{\sum_{x \in V} \xi(\beta(x), g(x))}{|V|}\right). \quad (1)$$

Por supuesto que existen, muchas otras formas de medir la calidad –en términos de la eficacia– de un clasificador. Por ejemplo en muchos problemas de la realidad, la abstención no tiene el mismo costo que un error, ni todos los errores el mismo costo, pues no es lo mismo decirle a un paciente con cáncer que está saludable que decirle a uno saludable que tiene cáncer. Por otro lado, cuando el problema admite que los objetos pertenezcan a varias clases, la valoración del error de clasificación es más compleja. La idea esencial aquí es que el problema de **la eficacia de los clasificadores no debe ser visto fuera del problema de la realidad que se enfrenta**.

Evaluar la eficacia de un clasificador con la misma muestra utilizada para entrenarlo no permite sacar conclusiones sobre su capacidad de generalización, pues se puede incurrir en el sobre-entrenamiento (el clasificador no comete errores con los objetos del conjunto de entrenamiento pero se equivoca con los nuevos). Para no correr este riesgo pueden ser usadas diversas técnicas [1], por ejemplo:

- Dividir la muestra que tenemos a disposición a la mitad y utilizar una para crear el clasificador y otra para evaluarlo (*Hold-out method*).
- Dividir la muestra en varios subconjuntos, promediar los resultados de la validación del clasificador con cada uno de los subconjuntos una vez que se usaron los restantes para definirlo (*Cross-validation*).

La meta de la clasificación supervisada es obtener clasificadores competentes a partir del conjunto de entrenamiento disponible. De hecho, la meta en la clasificación supervisada pudiera ser vista como la solución del problema de encontrar un clasificador que minimice (maximice) cierta *función de error* (eficacia) a partir de un conjunto de entrenamiento. Es importante, tener en cuenta la eficiencia computacional de los clasificadores (tanto el costo de entrenarlos, como el costo de realizar una clasificación) pues de nada sirve tener un clasificador que no cometa errores si tarda mucho tiempo en dar la respuesta. En este trabajo, no vamos a hacer un análisis profundo de la eficiencia, sin embargo, todos los algoritmos que se muestran han sido llevados a la práctica sin los mayores inconvenientes.

En el estudio de los algoritmos de clasificación supervisada, una herramienta que ha recibido mucha atención por parte de la comunidad científica es la descomposición del valor esperado –esperanza matemática– de una función de pérdida (*loss function*) a través de distintos conjuntos de entrenamiento,

en términos que permita estudiar más profundamente el comportamiento de estos algoritmos de manera general. Los términos en los que se suele descomponer dicho costo son: la desviación con respecto al cubrimiento subyacente, o sea el sesgo (*bias*) y la variación existente entre los clasificadores que produce el algoritmo de clasificación con distintos conjuntos de entrenamiento, o sea la varianza (*variance*), como se verá a continuación.

Los orígenes de la descomposición de los costos del error se encuentran en [6], aplicados en el contexto de la regresión supervisada, el cual es muy parecido al de la clasificación supervisada lo que en lugar de predecir las clases a las que pertenece un objeto se intenta estimar el valor de un número real. Supóngase la existencia de un universo de objetos U y un espacio de representación de dichos objetos S, además existe una función real f sobre S la cual se desea aproximar a partir de un conjunto de muestras de la forma $Z = \{(x,y)_i\}$ donde $y = f(x)$, o sea, se crea o utiliza un modelo $h(Z,x) \approx f(x)$, la idea es que h se aproxime lo mejor posible a f.

Un ejemplo de problema de regresión es el siguiente. Se desea predecir la altura de los hombres que viven en Cuba a partir de su peso, y ciudad de nacimiento. En este caso el universo U sería el conjunto de todos los hombres que viven en Cuba y el espacio de representación sería $S = (R_1 \times R_2)$ donde R_1 es el conjunto de los números reales (para representar el peso) y R_2 el conjunto de todas las ciudades de Cuba. La cuestión es a partir de una muestra de hombres, crear un modelo que sea capaz de predecir la altura de un nuevo hombre no presente en la muestra.

En el caso de la regresión supervisada una función de pérdida recibe dos números reales, el resultado del modelo y el resultado de la función (*ground-truth*), y devuelve el costo correspondiente. Utilizando la función de pérdida $\xi(x,y) = (x-y)^2$ –nótese como se puede aplicar una resta entre los parámetros de la función ya que estos son números reales–, la cual llamaremos en este trabajo *cuadrado de la diferencia*, se desea analizar cómo se comporta el modelo h de manera general, es decir, cómo se comporta utilizando distintas muestras de entrenamiento en un mismo punto x, se tiene entonces:

$$E_Z\left[\left(f(x) - h(Z,x)\right)^2\right] = E_z[(f(x) - E_z[h(Z,x)])^2] + E_Z[(E_Z[h(Z,x)] - h(Z,x))^2]. \qquad (2)$$

En la ecuación anterior se tiene la esperanza matemática de la función de costo del error a través de distintos conjuntos de entrenamientos descompuesta en la suma de un primer término que representa el sesgo (es decir, cuánto se diferencia el valor real de la función f del resultado que tiende a dar el modelo h a través de distintos conjuntos de entrenamiento) y un segundo que representa la varianza (es decir, cuánto varían las predicciones del modelo h para x a través de distintos conjuntos de entrenamiento). La gran ventaja que tiene esta descomposición es que es puramente aditiva, puede verse claramente que si se desea reducir el error hay que reducir o el sesgo o la varianza.

En el caso de la clasificación supervisada no ha sido posible obtener un resultado tan claro. Es importante notar que cuando la salida del modelo en cuestión no es un número real no se puede utilizar la diferencia del cuadrado como función de costo. Por ejemplo, suponiendo que los objetos pertenecen a una sola clase, en [7] se propone la siguiente descomposición del valor esperado de la función de pérdida en ceros y unos, donde se supone que cada objeto pertenece solamente a una clase:

$$E_Z\left[1 - I\left(x \in A(Z,x)\right)\right] = sesgo + varianza + ruido. \qquad (3)$$

Donde se tiene que:

$$sesgo = \left(\frac{1}{2}\sum_{i=1}^{k}(P(K_i|x) - P_z(A(Z,x) = K_i))^2\right),$$
$$varianza = (\frac{1}{2}(1 - \sum_{i=1}^{k}(P_z(A(Z,x) = K_i)^2)),$$
$$ruido = (\frac{1}{2}(1 - \sum_{i=1}^{k}(P(K_i|x))^2)).$$

El término $P_z(A(Z,x) = K_i)$ representa la probabilidad de que el algoritmo de clasificación A asigne la clase K_i al objeto representado por x a través de distintos conjuntos de entrenamiento. Por otra parte el término $(1 - I(x \in A(Z,x)))$ representa la función de costo en 0 y 1 donde I es una función indicadora. Esta descomposición ha sido criticada por el hecho de que el término llamado varianza no expresa una variación alrededor del valor estimado promedio [8]. Otras descomposiciones han sido propuestas en [9], [10], [11], [12], y todas tienen deficiencias notables. Pedro Domingos propone en [13] una descomposición unificada que contempla tanto el caso de regresión como el caso de clasificación, sin embargo para el caso de clasificación con función de costo en 0 y 1 dicha descomposición igualmente presenta sus problemas [1].

Aunque las descomposiciones de sesgo y varianza en el caso de la clasificación aún presentan problemas, estas han sido utilizadas para explicar diferentes fenómenos dentro del campo, como se verá en este trabajo.

2.3 Combinación de clasificadores

Dado que una de las tareas fundamentales en el área de la clasificación supervisada es mejorar la eficacia de los métodos existentes, surge la idea de combinar múltiples clasificadores de la misma forma en que intentamos buscar varias opiniones en los problemas que resuelven las personas a diario. De esta forma una *combinación de clasificadores* es un clasificador que basa su decisión en la decisión de otros clasificadores a los cuales nos referiremos como los *clasificadores individuales*. Según [14] en la literatura el término *esquema* (*ensemble*) es utilizado cuando los clasificadores individuales forman una familia, cuando los clasificadores no son de una misma familia entonces se suele utilizar el término de *sistemas de múltiples clasificadores* (*multiple classifiers systems)*.

El objetivo con el desarrollo de combinaciones de clasificadores es que estos tengan un mejor desempeño que los clasificadores individuales, o sea, presentar mejores soluciones a los problemas que no han sido resueltos de manera aceptable por los métodos tradicionales, entendiendo por métodos tradicionales a los clasificadores que no toman sus decisiones basados en otras decisiones anteriores. Por *desempeño de los clasificadores* nos referimos tanto a la eficacia como a la eficiencia de los mismos. Si los clasificadores individuales que van a ser combinados no cometen errores coincidentes en los puntos del espacio de representación, entonces la eficacia se puede mejorar si los resultados de estos se combinan adecuadamente. Por otra parte si se tiene un problema complejo, la eficiencia se puede mejorar dividiendo el problema en sub-problemas más simples, los cuales serán resueltos por los clasificadores individuales, el problema complejo se resuelve después integrando las soluciones dadas por los clasificadores individuales.

Dietterich en [15] se pregunta: ¿Por qué es posible construir una combinación donde los clasificadores no cometan los mismos errores? ¿Por qué no podemos encontrar un clasificador individual que trabaje tan bien o mejor que una combinación de clasificadores? A su vez propone tres razones por la cuales debe considerarse el uso de los esquemas de clasificadores, estas razones serán retomadas por Dietterich en [16] y Kuncheva en [1] y en este trabajo se expondrán a continuación.

La primera razón es estadística. En la práctica, una vez dado el conjunto de entrenamiento, la tarea es encontrar un clasificador lo más eficaz posible. Sin embargo, puede darse el caso de que en lugar de encontrar un solo clasificador que es el más eficaz, se obtengan varios con una eficacia igual de aceptable, pues todos tienen un error estimado muy semejante o igual. Sin embargo, estos clasificadores pueden generalizar de manera diferente, es decir, cuando se clasifican objetos no presentes en la muestra disponible los clasificadores no tienen por qué trabajar de la misma manera, estos pueden cometer errores sobre elementos distintos. Una opción disponible es tomar aleatoriamente uno de estos clasificadores, entonces el clasificador que se tome puede ser la mejor de las opciones, pero también la peor. Una alternativa puede ser tomarlos todos y de alguna manera promediar los resultados de estos para intentar reducir el riesgo de una clasificación incorrecta.

La segunda razón es computacional. En muchos casos los algoritmos de clasificación realizan una búsqueda en un conjunto de clasificadores posibles a partir del conjunto de entrenamiento en cuestión con el objetivo de encontrar un buen clasificador. En la literatura al conjunto de clasificadores posibles se le llama *conjunto de las hipótesis*. Realizar esta búsqueda de manera extensiva, o sea, tenerlos en cuenta a todos, puede ser muy costoso. Por ejemplo, el problema de encontrar el árbol de decisión más pequeño que es consistente con un conjunto de entrenamiento es NP-completo [17], también lo es el problema de encontrar los pesos de la red neuronal más pequeña consistente con un conjunto de entrenamiento [18]. Por esta razón, algunos algoritmos de clasificación realizan la búsqueda a través de alguna heurística, y esto trae como consecuencia la posibilidad de quedar atrapado en un óptimo local. Esto implica que en muchos casos aunque la información disponible acerca del problema –conjunto de entrenamiento– permita encontrar un clasificador con una alta eficacia, por problemas de complejidad computacional no será posible acceder a este. Entonces, si se toman varios clasificadores, –cada uno pudo haber sido un óptimo local distinto, por lo que no tienen por qué cometer los mismos errores– al combinarlos en búsqueda de consenso se puede lograr una mejor eficacia.

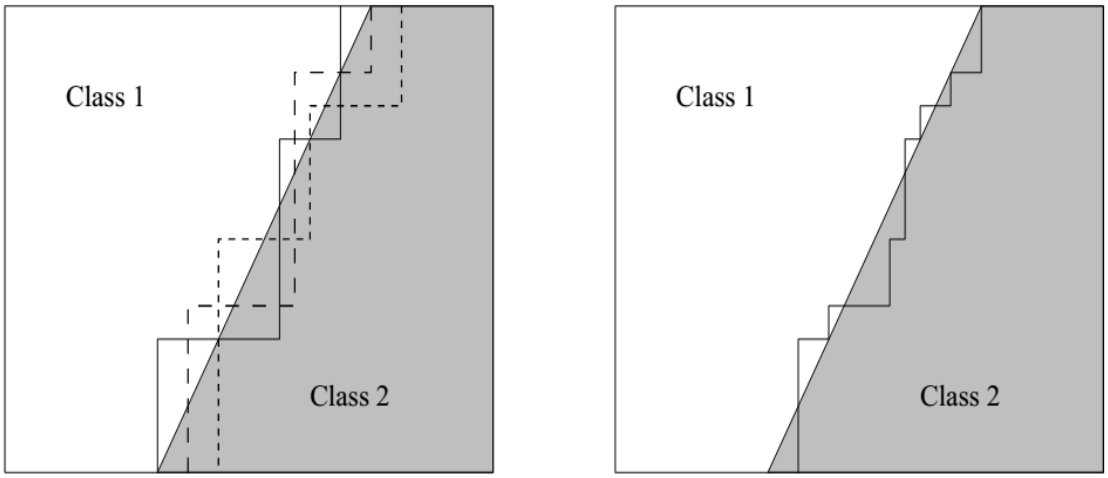

Fig. 1. : La imagen izquierda muestra el límite de decisión diagonal y las aproximaciones realizadas por varios árboles de decisión. La imagen a la derecha muestra el resultado de una aproximación a través de una votación realizada por las aproximaciones mostradas en la imagen de la izquierda.

La tercera razón es representacional. Como se ha visto anteriormente un algoritmo de clasificación define una familia de clasificadores. Existe la posibilidad de que el clasificador óptimo para un problema dado no se encuentre dentro de la familia de clasificadores que define el algoritmo de clasificación que se está utilizando. Por ejemplo, existen algoritmos de clasificación cuyos clasificadores discriminan entre dos clases a partir de un híper-plano entre estas; pero también existen problemas donde las clases se encuentran distribuidas de una forma más compleja, es decir, que no es posible separar las clases solamente a través de híper-planos. Entonces si se está obligado a considerar clasificadores dentro de la familia anterior solamente –quizás por razones de eficiencia, o simplicidad–, ocurre que el clasificador óptimo no está presente dentro de los posibles resultados del algoritmo de clasificación, pero si combinamos distintos clasificadores de este tipo entonces es posible un acercamiento mucho mayor al clasificador óptimo. Dietterich propone otro ejemplo: algunos de los clasificadores que utilizan un árbol de decisión generan una partición del espacio de representación –se asume R^n– donde las supuestas clases se delimitan a partir de híper-planos paralelos a los ejes de coordenadas, si el borde real que discrimina a dos clases de objetos es diagonal entonces el árbol de decisión aproximará este borde de una forma escalonada. Si combinamos distintos árboles, obtendremos distintas aproximaciones y mediante una votación se puede llegar a obtener una mejor aproximación, véase las figuras tomadas del trabajo de Dietterich (Figura 1). Nótese que este argumento puede ser rebatido, pues, que una familia de clasificadores no contenga a un clasificador óptimo para la

solución de un problema dado, no quiere decir que no exista otra familia de clasificadores que sí lo contenga. Sin embargo, es posible salvar el argumento de Dietterich, pues cabe la posibilidad de que obtener un clasificador de la segunda familia sea más complejo que obtener varios de la primera y combinarlos.

Es importante siempre tener en cuenta que una combinación de clasificadores es también un clasificador. En este trabajo la combinación de clasificadores será denotada por D, cada combinación cuenta con l clasificadores individuales los cuales serán denotados por D_i con $(1 \leq i \leq l)$.

3 Funciones de combinación de las salidas de los clasificadores

La manera en que se combinan los resultados de los clasificadores individuales es una cuestión fundamental en el estudio de las combinaciones de clasificadores. La idea aquí es optimizar la decisión que se va a tomar a partir de las decisiones individuales. Si todos los clasificadores individuales tienen el mismo espacio de salida, la salida de todos ellos para un objeto en específico puede ser codificada a través de una matriz DP que tiene k columnas y l filas. A la matriz DP se le llamará *matriz de perfil de decisión*. El valor de la matriz en la fila i y la columna j se denotará por $DP(i,j)$ y representa el valor de la componente número j de la tupla de salida del clasificador D_i para el objeto en cuestión, es decir, la fila i representa la tupla de salida del clasificador D_i y la columna j representa las salidas de los clasificadores respecto al subconjunto K_j.

Por ejemplo, la matriz de perfil de decisión que se muestra en la Tabla 1 corresponde al caso en que se tienen 5 clasificadores y 3 clases con las siguientes salidas para un objeto en cuestión:

- $D_1(x) = (0.5, 0.49, 0.1)$,
- $D_2(x) = (0.5, 0.49, 0.1)$,
- $D_3(x) = (0.5, 0.49, 0.1)$,
- $D_4(x) = (0, 0.9, 0.1)$,
- $D_5(x) = (0.1, 0.9, 0)$.

Se estudiarán a continuación distintas funciones de combinación de las salidas de los clasificadores. Es decir, en este epígrafe se asumirá que los clasificadores que componen la combinación ya han sido creados. Dichas funciones reciben como argumentos el objeto que se desea clasificar y las salidas de los clasificadores individuales para dicho objeto (codificadas a través de la matriz de perfil de decisión). Una función de combinación puede tener otros parámetros adicionales, como se verá más adelante.

Tabla 1. Ejemplo de matriz de perfil de decisión.

0.5	0.49	0.1
0.5	0.49	0.1
0.5	0.49	0.1
0	0.9	0.1
0.1	0.9	0

Es posible dividir las funciones de combinación según el tipo de información que necesitan de la salida de los clasificadores individuales [1], [5]. En este trabajo se seguirá el mismo convenio. Otra tendencia en la literatura [1] es la separación entre la combinación de clasificadores y la selección de clasificadores. Este último convenio no será seguido en este trabajo ya que ambos casos pueden ser contemplados a partir del concepto de función de combinación de salidas dada anteriormente, por consiguiente en este trabajo se estudiará la selección de clasificadores como un caso particular de función de combinación de las salidas de estos.

Es importante en este punto comentar acerca de la importancia de utilizar correctamente los tipos de salida que den los clasificadores individuales, ya que si se utiliza una función de combinación de salidas del tipo abstracto con clasificadores que tienen salidas del tipo medida se estaría desechando información, aunque esto pudiera ser conveniente en algunas aplicaciones. Por ejemplo, cuando las salidas obtenidas en la Tabla 1 son tratadas como resultados del tipo abstracto se tiene la matriz de perfil de decisión mostrada en la Tabla 2.

Como resultado de esta conversión se ignora que aunque los tres primeros clasificadores dan como subconjunto correcto al primero –o sea K_1, pues es el que recibe un mayor soporte–, también dan un soporte muy semejante al subconjunto K_2 y que los dos últimos clasificadores dan un soporte muy bajo al primero y sin embargo uno muy alto al segundo. Es decir, utilizando la salida de tipo medida se puede observar que de manera general hay mucho más soporte para la segunda clase que para la primera, sin embargo cuando se convierte a tipo abstracto no sucede así.

Tabla 2. Ejemplo de matriz de decisión con salidas abstractas.

1	0	0
1	0	0
1	0	0
0	1	0
0	1	0

A continuación se expondrán distintas funciones de combinación de salidas de los clasificadores individuales. Se comenzará con las funciones que combinan salidas de tipo abstracto, luego se continuará con las funciones que combinan salidas de tipo rango y medida, por ese orden. Al final se tendrá en cuenta la selección de clasificadores. Para los métodos de combinación de clasificadores que se expondrán a continuación, se asumirá que las clases forman una partición sobre el universo, en caso contrario se hará la aclaración explícita.

3.1 Combinación de salidas de tipo abstracto

En este epígrafe se estudiarán distintos métodos de combinación que tienen en cuenta salidas de tipo abstracto.

3.1.1 Algoritmos basados en votación

Los algoritmos basados en votación consideran la salida de cada clasificador como un voto. Entre estos se encuentran los denominados: *voto mayoritario* y *voto ponderado*.

Votación mayoritaria

En el caso del voto mayoritario a partir de todos los votos emitidos se trata de llegar a cierto tipo de *consenso*. Entre las formas de consenso más manejadas se encuentran la *unanimidad*, la *simple mayoría* y la *pluralidad*.

En el consenso por unanimidad se exige que todos los clasificadores estén de acuerdo respecto a una misma clase K_i, es decir, todos deben tener la misma salida, en caso contrario la combinación se abstiene de clasificar. En otras palabras, el resultado de la combinación sería K_j si $\sum_{i=1}^{l} DP(i,j) = l$, y una abstención en otro caso.

En el consenso por simple mayoría se exige que más de la mitad de los clasificadores concuerden en una clase, o sea, el resultado de la combinación es K_j si $\sum_{i=1}^{l} DP(i,j) > \left\lfloor {}^{l}/_{2} \right\rfloor$, y una abstención en otro caso.

En el consenso por pluralidad se toma como resultado la clase por la que más se vota. Es decir, se tiene que K_j es el resultado final de la combinación si se verifica que $\sum_{i=1}^{l} DP(i,j) = max_{j=1}^{k}\{ \sum_{i=1}^{l} DP(i,j) \}$ los casos de empates pueden ser resueltos arbitrariamente o desembocar en una abstención.

En todos los casos la salida de la combinación no es el subconjunto K_j, sino una tupla con ceros en todas las posiciones excepto en la posición número j en la cual va un uno.

El voto mayoritario es el término utilizado en la literatura para referirse a los algoritmos basados en votación más simples, usualmente se define el tipo de consenso que se va a utilizar y después se utiliza solamente dicho término. La simple mayoría es el método de consenso más estudiado en la literatura [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29] y [30]. Algunos de los resultados en torno a esta forma de consenso, por su interés, son expuestos a continuación.

Si se asume que todos los clasificadores individuales en la combinación tienen una misma probabilidad p de clasificar correctamente a un objeto dado, y que cada uno de ellos toma su decisión independientemente de la decisión que tomen los demás, entonces la probabilidad de clasificación correcta de la combinación, si se utiliza el voto mayoritario, puede ser expresada mediante la ley binomial, mostrada en (4**)**.

$$\sum_{j=\lfloor l/2 \rfloor}^{l} \binom{l}{j} p^j (1-p)^{l-j}, \quad (4)$$

En la ecuación anterior el término $\binom{l}{j} p^j (1-p)^{l-j}$ representa la probabilidad de que exactamente j clasificadores hayan votado de manera correcta y por consiguiente $(l-j)$ de manera incorrecta. Cuando se dice que el clasificador D_i es independiente con respecto a D_j, se asume que para cualquier objeto en U representado por x, y cualesquiera a, b tomados de los espacios de salida de D_i y D_j respectivamente, se cumple que $P(D_i(x) = a) = P\big(D_i(x) = a \big| D_j(x) = b\big)$, es decir, la probabilidad de que D_i dé un resultado no se afecta una vez que se conoce el resultado dado por D_j. Todos los clasificadores en un conjunto son independientes entre sí, si para cualquier combinación posible que se tome de ellos estos permanecen independientes.

Teorema de Condorcet

Entre los resultados teóricos que respaldan al voto mayoritario se encuentra el teorema de Condorcet [31], el cual fue publicado en el año 1785 y asume que se tiene que tomar una decisión que tiene dos alternativas $(+/-)$, una es correcta $(+)$ y la otra es incorrecta $(-)$ y que tenemos l individuos que tomarán la decisión independientemente, y estas serán combinadas usando el voto mayoritario para llegar a una solución final. Entonces…

Teorema 1 (de Condorcet). Si cada individuo hace su voto independientemente de los votos de los demás, y cada voto tiene una probabilidad $p > 0.5$ de ser correcto entonces cuando la cantidad de individuos tiende a infinito, la probabilidad de que la decisión del voto mayoritario sea correcta es una función monótona creciente y tiende a 1. En cambio, si $p < 0.5$, entonces la probabilidad del consenso correcto, a través del voto mayoritario, es una función monótona decreciente y tiende a 0. Si $p = 0.5$ entonces la probabilidad del consenso correcto se mantiene en 0.5.

Este teorema da soporte a la idea intuitiva de que si juntamos muchos individuos que tienen mayor probabilidad de tomar la decisión correcta que de equivocarse, entonces podemos esperar que un consenso entre estos tenga muchas posibilidades de ser correcto.

En el contexto de la combinación de clasificadores supervisados, cada individuo representa un clasificador que puede votar por el K_i correcto o equivocarse. Sin embargo, este teorema no ofrece grandes asideros ya que solo es posible estimar la probabilidad de que un clasificador vote correctamente a partir de un conjunto de validación, nótese que puede ocurrir que el conjunto de entrenamiento disponible no sea lo suficientemente representativo como para permitir una buena generalización. Además que l tienda a infinito es imposible de lograr en la práctica. Lo que con un espíritu muy pragmático, tendría que traducirse en que la cantidad de clasificadores a combinar sea muy

grande para que el teorema pueda dar algún aval al resultado. Esto conlleva un inconveniente que no podemos olvidar y es el costo computacional de tal solución.

Análisis de Louisa Lam y Ching Suen

En el año 1997 Lam y Suen [30] realizan un estudio del voto mayoritario donde obtienen algunos resultados interesantes. Estos autores estudiaron el voto mayoritario como método de combinación de un número par o impar de clasificadores que pueden votar por una de las k clases o abstenerse.

Los autores llegaron a resultados que permiten conocer, al menos teóricamente, qué se puede esperar de una combinación cuando estas tienen un número par o impar de clasificadores, o sea, permiten discriminar en qué situaciones es preferible tener una cantidad par de clasificadores. También obtuvieron resultados relajando la restricción de que todos los clasificadores tengan una misma probabilidad p de tomar la decisión correcta, más específicamente, matemáticamente evalúan el efecto que puede tener agregar nuevos clasificadores a la combinación. Un resumen de todos los resultados alcanzados en este trabajo será expuesto a continuación.

Sean PC y PW las probabilidades de un consenso –utilizando simple mayoría– correcto o incorrecto, respectivamente. Nótese que en el problema más simple –no abstenciones, número impar de clasificadores y dos clases– $PC + PW = 1$, sin embargo en el caso que se está estudiando esto no se verifica debido a que el voto mayoritario tiene la opción de abstenerse. Denotaremos por $PC(l)$ y $PW(l)$ a las probabilidades de consenso utilizando l clasificadores. Inicialmente los referidos autores demuestran los siguientes teoremas:

Teorema 2. Si los clasificadores de la combinación son independientes y todos ellos tienen una misma probabilidad (denotada por p) de clasificar correctamente a un elemento, entonces se verifican las siguientes ecuaciones:

1- $PC(2n+1) = PC(2n) + p^{n+1}(1-p)^n\binom{2n}{n}$. (5)

2- $PC(2n) = PC(2n-1) - p^n(1-p)^n\binom{2n-1}{n}$. (6)

Corolario 2.1. $PC(2n+1) - PC(2n-1) = p^n(1-p)^n\binom{2n-1}{n}(2p-1)$.

Corolario 2.2. $PC(2n+2) - PC(2n) = p^{n+1}(1-p)^n\binom{2n}{n}\left[\frac{2np+p-n}{n+1}\right]$.

Corolario 2.3. $PC(2n+2) - PC(2n-1) = p^n(1-p)^n\binom{2n}{n}\left[\frac{(4n+2)p^2-2np-(n+1)}{2(n+1)}\right]$.

Los resultados anteriores se obtienen a partir de (4**)** a través de simple trabajo algebraico. De los resultados anteriores son derivadas, entre otras, las consecuencias siguientes:

- $PC(2n) < PC(2n+1)$ y $PC(2n) < PC(2n-1)$ para todo n y p. Nótese que en la primera ecuación del Teorema 2 $PC(2n+1)$ es igual a $PC(2n)$ más un número positivo, y que en la segunda ecuación $PC(2n)$ es igual a $PC(2n-1)$ menos un número positivo. Esto es, bajo las condiciones expuestas, se tiene que cuando se combina un número par de clasificadores la probabilidad de un consenso correcto es menor que cuando se hace la combinación con un clasificador menos o un clasificador más. Esto es interesante ya que se ve cómo el crecimiento o decrecimiento de la función va haciendo un zigzag.
- Cuando se combina un número par de clasificadores se tiene que $PC(2n)$ es monótona creciente si se tiene que $p > (n/(2n+1))$, lo que se verifica siempre que $p \geq 0.5$, esto es, según se van agregando dos clasificadores a la combinación PC crece. En cambio, si se tiene que $p < (n/(2n+1))$, lo que se verifica siempre que $p < 1/3$, entonces $PC(2n)$ es monótona decreciente. Si $(1/3) \leq p < (1/2)$ entonces $PC(2n)$ se comportará dependiendo de los valores de las magnitudes p y $n/(2n+1)$. Esto se debe al Corolario 2.2 que establece la relación que tiene la diferencia entre $PC(2n)$ y $PC(2n+2)$ con n y p.

- Teniendo en cuenta valores pares e impares, y desarrollando el Corolario 2.3 se verifica que $PC(2n+2) > PC(2n-1)$ si y solo si $p > f_1(n) = (n + \sqrt{5n^2 + 6n + 2}/4n + 2)$, en cambio si $p < f_1(n)$ se invierte la desigualdad. Como se tiene que $f_1(n)$ tiende a $p_u \approx 0.8090$ cuando n tiende a infinito, podemos decir que si $p \geq p_u$ entonces $PC(2n+2) > PC(2n-1)$.
- Cuando $p \geq p_u$ se puede establecer un orden para PC y es el siguiente:
  $$PC(2n) < PC(2n-1) < PC(2n+2) < PC(2n+1) < PC(2n+4) < \cdots$$
  Por ejemplo: $PC(2) < PC(1) < PC(4) < PC(3) < PC(6) < PC(5) < PC(8) < \cdots$
- $PC(2n)$ tiende al mismo límite que $PC(2n-1)$ para todos los valores posibles de p en $(0,1)$.

En su trabajo Lam muestra una tabla (Tabla 3) con las probabilidades del consenso ya calculadas para distintos valores de p y n. En dicha tabla es posible observar (verificar) los comportamientos de PC según los valores de p y n.

Además, los resultados antes planteados fueron verificados a través de experimentos realizados por los autores del mencionado trabajo. Básicamente la idea fue tomar un conjunto de clasificadores (6 clasificadores) para combinar con el voto mayoritario a varios subconjuntos de estos, medir los desempeños y compararlos. Por ejemplo, resultó que cuando se combinaron grupos de 3 clasificadores, el desempeño fue mejor que cuando se combinaron grupos de 4. La verificación no resultó exacta, es decir, no se verificó que todos las combinaciones de 3 fueran mejores que todas las combinaciones de 4, lo cual es lógico, ya que en la práctica no se puede lograr que los clasificadores cumplan las restricciones antes expuestas. Sin embargo, se pudo notar una tendencia bastante marcada. La conclusión más importante que se puede extraer de los resultados anteriores es que cuando se combina un número par de clasificadores disminuye la probabilidad de consenso correcto, pero también la de consenso incorrecto, por lo que aumenta la probabilidad de abstención, y en problemas donde el costo de una equivocación es mucho mayor que el costo de una abstención estaría justificado el uso de un número par de clasificadores.

El modelo propuesto por Lam para el estudio del voto mayoritario tiene grandes limitaciones en la práctica. Trabaja con la probabilidad de que un clasificador etiquete correctamente a un objeto, la cual se puede estimar, pero siempre se corre el riesgo de que la muestra disponible de objetos no generalice bien el problema. Además asumir que todos los clasificadores tienen la misma eficacia es una condición muy fuerte para ser asumida en la práctica. Por otro lado se hace necesario que el número de clasificadores a combinar sea muy grande para que los resultados expuestos avalen de alguna manera su uso.

Para enfrentar el problema creado al asumir que los clasificadores tienen la misma probabilidad de etiquetar correctamente a un nuevo objeto, los mencionados autores presentan un modelo para relajar esta condición. En particular se proponen medir la diferencia entre la eficacia de una combinación de clasificadores con distintos p_i –esto es la probabilidad de que el i-ésimo clasificador asigne un objeto al K_i correcto– y la nueva combinación que se crea al añadir uno o dos clasificadores nuevos. El análisis solo tiene sentido cuando los nuevos votos cambian el resultado de la combinación anterior, es decir, los nuevos votos producen una diferencia. Estos casos son mostrados en la Tabla 4.

Sin asumir nada acerca de las probabilidades de los clasificadores, ni acerca de la independencia entre estos, se puede notar que cuando se agrega un solo voto a una combinación con $2n$ clasificadores, se reduce la probabilidad de abstención ya que las diferencias solo ocurren cuando anteriormente había un empate y el nuevo voto lo rompe, no importa si es para crear un consenso correcto o incorrecto; y que en cambio cuando se agrega un nuevo voto a una combinación que tiene un número impar de clasificadores ocurre lo contrario, ya que las diferencias ocurren cuando anteriormente existía una decisión, ya fuera esta un consenso correcto o incorrecto, y el nuevo voto forma un empate.

Es muy confuso analizar el efecto de añadir dos clasificadores a una combinación ya que el efecto producido por el segundo parece contrarrestar el producido por el primero. Según Lam, cuando se añaden 2 votos a un número par de clasificadores las diferencias pueden ocurrir de dos formas distintas, se puede pasar de una abstención a una clasificación correcta, pero también se puede pasar de una clasificación correcta a una abstención. ¿Cuál de los dos casos de diferencias debe ocurrir más

frecuentemente? Asumiendo la independencia entre los clasificadores y denotando las probabilidades de clasificación correcta de los dos que se agregan por q_1 y q_2, calcularon que la probabilidad de pasar de una abstención a un consenso correcto es mayor que la probabilidad de pasar de un consenso correcto a una abstención si se verifica que:

$$\frac{q_1 q_2}{(1-q_1)(1-q_2)} \geq \frac{p_i}{(1-p_i)} \ \forall i, 1 \leq i \leq 2n.$$

Análogamente ocurre para el caso de añadir 2 votos a un número impar de clasificadores, en este caso para incrementar la probabilidad de consenso correcto es necesario que se verifique:

$$\frac{q_1 q_2}{(1-q_1)(1-q_2)} > \frac{p_i}{(1-p_i)} \ \forall i, 1 \leq i \leq 2n+1.$$

Tabla 3. Tabla mostrada por Lam en su trabajo que refleja los valores que toma $PC(n)$ para distintos valores de p y n que representan la probabilidad de cada clasificador de clasificar correctamente un objeto y el número de clasificadores que se combinan respectivamente.

p	Values of n								
	2	3	4	5	6	7	8	9	10
0.10	0.0100	0.0280	0.0037	0.0086	0.0013	0.0027	0.0004	0.0009	0.0001
0.15	0.0225	0.0608	0.0120	0.0266	0.0059	0.0121	0.0029	0.0056	0.0014
0.20	0.0400	0.1040	0.0272	0.0579	0.0170	0.0333	0.0104	0.0196	0.0064
0.25	0.0625	0.1562	0.0508	0.1035	0.0376	0.0706	0.0273	0.0489	0.0197
0.30	0.0900	0.2160	0.0837	0.1631	0.0705	0.1260	0.0580	0.0988	0.0473
0.35	0.1225	0.2818	0.1265	0.2352	0.1174	0.1998	0.1061	0.1717	0.0949
0.40	0.1600	0.3520	0.1792	0.3174	0.1792	0.2898	0.1737	0.2666	0.1662
0.45	0.2025	0.4253	0.2415	0.4069	0.2553	0.3917	0.2604	0.3786	0.2616
0.50	0.2500	0.5000	0.3125	0.5000	0.3438	0.5000	0.3633	0.5000	0.3770
0.55	0.3025	0.5748	0.3910	0.5931	0.4415	0.6083	0.4770	0.6214	0.5044
0.60	0.3600	0.6480	0.4752	0.6826	0.5443	0.7102	0.5941	0.7334	0.6331
0.65	0.4225	0.7183	0.5630	0.7648	0.6471	0.8002	0.7064	0.8283	0.7515
0.70	0.4900	0.7840	0.6517	0.8369	0.7443	0.8740	0.8059	0.9012	0.8497
0.75	0.5625	0.8438	0.7383	0.8965	0.8306	0.9294	0.8862	0.9511	0.9219
0.80	0.6400	0.8960	0.8192	0.9421	0.9011	0.9667	0.9437	0.9804	0.9672
0.85	0.7225	0.9393	0.8905	0.9734	0.9527	0.9879	0.9786	0.9944	0.9901
0.90	0.8100	0.9720	0.9477	0.9914	0.9842	0.9973	0.9950	0.9991	0.9984
0.95	0.9025	0.9928	0.9860	0.9988	0.9978	0.9998	0.9996	1.0000	0.9999

De forma análoga se analizan las posibilidades para encontrar variaciones en el voto mayoritario, por ejemplo, si tenemos un número impar de clasificadores y se desea aumentar la fiabilidad de la combinación –disminuir la probabilidad del consenso equivocado– podemos lograrlo de dos modos distintos, eliminando un clasificador o duplicando el voto de uno de los ya presente –por duplicar el voto se entiende que el voto de uno de los clasificadores vale doble–. En este sentido el trabajo de Lam intenta explicar cuál de las dos opciones, debe brindarnos un mejor resultado, tanto para el caso en que se cuenta con un número par de clasificadores como para cuando se cuenta con un número impar. Otros análisis y resultados están presentes en el trabajo de Lam pero no es objetivo nuestro agotarlos todos aquí.

Tabla 4. : Casos donde se afecta la decisión de la combinación por motivo de agregar nuevos clasificadores.

Caso	Votos originales	Nuevos votos	Decisión original	Nueva decisión
Efecto cuando se tienen $2n$ votos y se añade 1 voto.				
1	n correctos, n incorrectos	1 correcto	Abstención	Correcta
2	n incorrectos por K_i, n correctos o incorrectos distintos de K_i	1 incorrecto por K_i	Abstención	Equivocada
Efecto cuando se tienen $2n+1$ votos y se añade 1 voto.				
3	$n+1$ correctos, n incorrectos	1 incorrecto.	Correcta	Abstención
4	$n+1$ incorrectos por K_i, n correctos o incorrectos distintos de K_i	1 distinto de K_i	Equivocada	Abstención
Efecto cuando se tienen $2n$ votos y se añaden 2 votos.				
5	n correctos, n incorrectos	2 correctos	Abstención	Correcto
6	$n+1$ correctos, $n-1$ incorrectos	2 incorrectos	Correcto	Abstención
Efecto cuando se tienen $2n+1$ votos y se añaden 2 votos.				
7	n correctos, $n+1$ incorrectos	2 correctos	Abstención o Equivocada	Correcto
8	$n+1$ correctos, n incorrectos	2 incorrectos	Correcto	Abstención o Equivocada

En general, los resultados alcanzados por Lam y Suen representan una teoría acerca del comportamiento del voto mayoritario en diversas circunstancias. El principal problema es que se trabaja con la probabilidad de que un clasificador trabaje correctamente, la cual se puede estimar de manera incorrecta si la muestra de elementos disponibles no generaliza de forma adecuada el problema en cuestión.

Patrón de éxito y patrón de fracaso

El patrón de éxito y el patrón de fracaso fueron introducidos en [27] para ilustrar los límites del voto mayoritario. Supongamos que tenemos l clasificadores, con la misma probabilidad p de clasificar correctamente a un objeto dado, esto es, si tenemos 10 objetos en U y dicha probabilidad es $p = 0.6$ entonces cada clasificador etiqueta correctamente a solamente 6 de ellos. En este punto, dados la probabilidad p y la cantidad de objetos, se pueden generar todos los posibles resultados de los clasificadores para todos los objetos, entendiendo por resultado si el clasificador se equivoca o no. Para los parámetros anteriores –10 objetos, $p = 0.6$– Kuncheva construye la Tabla 5 la cual muestra todos

los posibles resultados de 3 clasificadores, y además muestra la eficacia del voto mayoritario en cada caso.

La Tabla 5 tiene 10 columnas, la primera es para numerar los resultados, las 8 siguientes codifican las posibles combinaciones de los resultados de los clasificadores –las combinaciones están representadas por una tupla de ceros y unos, el uno en la posición número i significa que el clasificador individual número i obtuvo una respuesta correcta–, la última columna muestra la eficacia del voto mayoritario en cada caso. Las filas de la Tabla 5 representan resultados posibles, el número en cada celda representa la cantidad de objetos que registraron la combinación de la columna correspondiente.

Por ejemplo, un resultado posible sería cuando las salidas de los 3 clasificadores son equivalentes, o sea, los 3 votan correctamente por los mismos 6 objetos y los 3 se equivocan en los 4 objetos restantes, en ese caso la columna que representa la combinación $(1, 1, 1)$ debe tener un 6 en la fila que represente tal resultado posible y la columna que representa la combinación $(0, 0, 0)$ debe tener un 4 en la misma fila, tal resultado es mostrado en la entrada 28 de la Tabla 5.

Tabla 5. Todas las posibles combinaciones de los votos de 3 clasificadores con $p = 0.6$ y 10 objetos.

No.	$(1,1,1)$	$(1,0,1)$	$(0,1,1)$	$(0,0,1)$	$(1,1,0)$	$(1,0,0)$	$(0,1,0)$	$(0,0,0)$	PC
1	0	2	2	2	4	0	0	0	0.8
2	0	2	3	1	3	1	0	0	0.8
3	0	3	3	0	3	0	0	1	0.9
4	1	1	1	3	4	0	0	0	0.7
5	1	1	2	2	3	1	0	0	0.7
6	1	2	2	1	2	1	1	0	0.7
7	1	2	2	1	3	0	0	1	0.8
8	2	0	0	4	4	0	0	0	0.6
9	2	0	1	3	3	1	0	0	0.6
10	2	0	2	2	2	2	0	0	0.6
11	2	1	1	2	2	1	1	0	0.6
12	2	1	1	2	2	1	1	0	0.7
13	2	1	2	1	2	1	0	1	0.7
14	2	2	2	0	2	0	0	2	0.8
15	3	0	0	3	2	1	1	0	0.5
16	3	0	0	3	3	0	0	1	0.6
17	3	0	1	2	1	2	1	0	0.5
18	3	0	1	2	2	1	0	1	0.6
19	3	1	1	1	1	1	1	1	0.6
20	3	1	1	1	2	0	0	2	0.7
21	4	0	0	2	0	2	2	0	0.4
22	4	0	0	2	1	1	1	1	0.5
23	4	0	0	2	2	0	0	2	0.6
24	4	0	1	1	1	1	0	2	0.6
25	4	1	1	0	1	0	0	3	0.7
26	5	0	0	1	0	1	1	2	0.5
27	5	0	0	1	1	0	0	3	0.6
28	6	0	0	0	0	0	0	4	0.6

Son destacables en dicha tabla –Tabla 5–, dos posibles resultados muy particulares, aquel en la que la eficacia del voto mayoritario registra la mejora más grande sobre las eficacias individuales (o sea p), es decir, la fila número 3 de la tabla, y aquel donde se produce el empeoramiento más grande, es decir,

la fila número 21 de la tabla. A dichos resultados se les llamará *patrón de éxito* y *patrón de fracaso* respectivamente. Kuncheva da definiciones formales y generales de ambos.

Sean los conjuntos $Q = \{0,1\}^l$ y $\mathbb{N}$ el conjunto de los números naturales. Los elementos en Q codifican las salidas de los l clasificadores en correctas o no, el número 1 en la posición i significa que D_i clasificó correctamente. O sea, por cada objeto a clasificar, a partir de la matriz de perfil de decisión de los clasificadores se obtiene un elemento de Q siempre que se conozca el K_i al que pertenece el objeto en cuestión.

Un resultado posible de los clasificadores sobre un conjunto de objetos V es una función $f: Q \to \mathbb{N}$ que codifica la cantidad de ocurrencias de cada elemento de Q sobre las salidas de los clasificadores por cada elemento de V.

Entiéndase, para la fila 28 de la tabla anterior se tiene $f(1,1,1) = 6$, $f(0,0,0) = 4$ y para cualquier otra combinación de ceros y unos f es igual a cero.

Sea $Q_i \subset Q$, el subconjunto de Q formado por todos los elementos que tienen un 1 en la posición número i. Sea $f_i = \sum_{q \in Q_i} f(q)$, es decir, f_i representa la cantidad de elementos de V que el clasificador D_i clasifica correctamente. Si todos los clasificadores tienen una misma probabilidad p de clasificar correctamente un elemento del conjunto V, entonces $\forall i[(f_i/|V|) = p], 1 \leq i \leq l$.

Sea la función $unos: Q \to \mathbb{N}$, que codifica la cantidad de unos que tiene un elemento de Q. Y sea la función $ceros: Q \to \mathbb{N}$ que codifica la cantidad de ceros.

Definición 1 [27]**.** El *patrón de éxito* es el siguiente resultado posible de los clasificadores sobre un conjunto V de objetos a clasificar:

$$f(q) = \begin{cases} \alpha, unos(q) = (\lfloor l/2 \rfloor + 1) \wedge ceros(q) = \lfloor l/2 \rfloor \\ \Upsilon, ceros(q) = l \\ 0, en\ otro\ caso \end{cases}, \qquad (7)$$

O sea, el patrón de éxito cumple las siguientes propiedades:

1- α es la cantidad de veces que ocurre cualquier combinación de exactamente $\lfloor l/2 \rfloor + 1$ votos correctos.
2- Υ es la cantidad de veces que todos los votos son incorrectos.
3- No ocurre otra distribución de los votos.

De esta forma α y Υ son variables que dependen de l y p. La idea con este resultado posible es no desperdiciar votos correctos. Es importante notar que el voto mayoritario da el mismo resultado correcto cuando se tienen exactamente $\lfloor l/2 \rfloor + 1$ votos por la clase correcta, que cuando se tienen exactamente $(\lfloor l/2 \rfloor + 1) + m$, con m entre 1 y $(l - (\lfloor l/2 \rfloor + 1)$, por lo que en el segundo caso se están desperdiciando exactamente m votos correctos ya que dichos votos no hacen la diferencia y cada clasificador tiene un número restringido por p de votos correctos sobre V. Por esta razón el patrón de éxito no deja que ocurran consensos correctos con $m > 0$. Además, en los consensos incorrectos todos los clasificadores se equivocan, pues sino se estarían desperdiciando votos correctos igualmente, o sea, votos correctos que no sirvieron para nada.

La probabilidad del consenso correcto del voto mayoritario es la suma de las probabilidades de las combinaciones en las que el voto mayoritario es correcto, o sea:

$$PC(l) = \frac{\sum_{q \in Q | unos(q) > \lfloor l/2 \rfloor + 1} f(q)}{|V|}.$$

La ecuación anterior cuenta la cantidad de veces que $\lfloor l/2 \rfloor + 1$ o más clasificadores dieron la respuesta correcta y después divide por la cantidad de objetos en V. Por ejemplo en la fila 1 de la Tabla 5sería $PC(3) = (0 + 2 + 2 + 4)/10$. Para el patrón de éxito la probabilidad de consenso correcto también se puede escribir como:

$$PC(l) = \frac{\binom{l}{\lfloor l/2 \rfloor + 1} * \alpha}{|V|}.$$

Donde el primer término del producto en el numerador es la cantidad de veces que se pueden sacar subconjuntos de tamaño $\lfloor l/2 \rfloor + 1$ –que serían los votos correctos– de un conjunto de tamaño l –todos los votos–, y el segundo término es, según la Definición 1, la cantidad de veces que ocurre cualquier combinación de $\lfloor l/2 \rfloor + 1$ votos correctos y $\lfloor l/2 \rfloor$ votos incorrectos. Después de varias sustituciones algebraicas se obtiene que en el patrón de éxito se cumple que $PC(l) = pl/(\lfloor l/2 \rfloor + 1)$ siendo la probabilidad $p \leq (\lfloor l/2 \rfloor + 1)/l$, si $p = (\lfloor l/2 \rfloor + 1)/l$ entonces $PC(l) = 1$.

Nótese que si $p > (\lfloor l/2 \rfloor + 1)/l$ no se puede obtener el patrón de éxito pues aunque se alcance $PC(l) = 1$, igual hay que desperdiciar votos obligatoriamente, es decir, el patrón de éxito es acerca de la mayor mejora sobre p, no acerca de la mejor $PC(l)$.

Definición 2. El *patrón de fracaso* es el siguiente resultado posible de los clasificadores sobre un conjunto V de objetos a clasificar:

$$f(q) = \begin{cases} \beta, ceros(q) = (\lfloor l/2 \rfloor + 1) \wedge unos(q) = \lfloor l/2 \rfloor \\ \mu, unos(q) = l \\ 0, en\ otro\ caso \end{cases} . \qquad (8)$$

El patrón de fracaso es una combinación que cumple las siguientes propiedades:

1- β es la cantidad de veces que ocurre cualquier combinación de $\lfloor l/2 \rfloor + 1$ votos incorrectos y $\lfloor l/2 \rfloor$ votos correctos.
2- μ es la cantidad de veces que todos los votos son correctos.
3- No ocurre otra distribución de los votos.

Para el patrón de fracaso se tiene que $PC(l) = (pl - \lfloor l/2 \rfloor)/(\lfloor l/2 \rfloor + 1)$ y si $p > 0.5$ este siempre es posible. Contrario al patrón de éxito la idea del patrón de fracaso es desperdiciar la mayor cantidad de votos correctos posibles, así, cuando se tenga asegurada una clasificación incorrecta con solo $\lfloor l/2 \rfloor + 1$ votos incorrectos los votos restantes son desperdiciados votando correctamente, y cuando el voto mayoritario dé como resultado una clasificación correcta igual lo hará con todos los votos correctos posibles.

De este estudio Kuncheva [27] concluye que la independencia entre los clasificadores no es la mejor situación que se puede tener, el patrón de éxito es mejor. Otra forma de expresar esto es que si el voto mayoritario va a ser usado, en lugar de trabajar por obtener clasificadores independientes es más provechoso trabajar por obtener clasificadores que registren el patrón de éxito. Nótese que igual un conjunto de clasificadores independientes pueden registrar el patrón de éxito, o sea, lo que se propone es trabajar directamente por el patrón de éxito. Aunque tanto la independencia como el patrón de éxito son condiciones imposibles de alcanzar en la práctica, el estudio de estas cuestiones sirve para lograr un mayor entendimiento acerca de cómo funcionan las combinaciones de clasificadores que utilizan el voto mayoritario, se logra que los investigadores se hagan una idea de las fuerzas que actúan en el problema, aunque no las puedan controlar.

Nótese que estas formas de dependencias entre los clasificadores también pueden traer malos resultados, por ejemplo en el patrón de fracaso, donde se registra el mayor empeoramiento de la eficacia con respecto a los clasificadores individuales. Es poco realista asumir que todos los clasificadores tienen la misma eficacia, sería interesante estudiar una generalización del patrón de éxito y del patrón de fracaso para el caso más general, es decir, donde no todos los clasificadores tengan la misma probabilidad de clasificar correctamente a un nuevo objeto.

Votación ponderada

Si los clasificadores en la combinación no tienen la misma eficacia, resulta lógico dar un peso mayor en la votación a aquellos de los cuales se espera un mejor desempeño. Por ejemplo, a partir de la matriz de perfil de decisión es posible definir la siguiente familia de funciones como funciones discriminativas de cada clase:

$$g_j = \sum_{i=1}^{l} b_i * DP(i,j). \quad (9)$$

En la ecuación anterior el término b_i es el peso correspondiente al clasificador D_i, y por supuesto que la cuestión está en hallar (ajustar) dichos pesos de manera que la eficacia de la combinación sea lo mayor posible. Existen sobrados ejemplos [1], [32] de casos en los que la asignación correcta de los pesos a los clasificadores conlleva a una mejora de la eficacia del voto mayoritario tradicional y de la eficacia del mejor clasificador individual.

Los autores en [1] y [32] recomiendan usar $b_i = \log(p_i/(1-p_i))$ donde p_i representa la probabilidad de clasificación correcta del clasificador D_i y debe ser estimada a partir del conjunto de entrenamiento. Dichos autores se basan en un resultado que demuestran para el caso de un problema con 2 clases solamente y que según [1] fue desarrollado independientemente en distintas áreas de la ciencia de la computación aunque parece ser que su primera aparición fue en [33].

El problema con este método de combinación es que una mala estimación de las probabilidades p_i puede llevar a un empeoramiento muy grande de la eficacia con respecto a los clasificadores individuales y al voto mayoritario tradicional.

3.1.2 Enfoque probabilístico

Sea $d = (d_1, d_2, \dots, d_l)$ una tupla que codifica las salidas de los l clasificadores que van a ser combinados, $d_j = K_i$ si el clasificador D_j asigna el objeto representado por x al subconjunto K_i. Entonces en el enfoque probabilístico se estiman las probabilidades de los subconjuntos K_i una vez conocidas las salidas de los clasificadores individuales –o sea la tupla d–, esto es $P(K_i|d)$.

Naïve Bayes

Aplicando la regla de Bayes y asumiendo que los clasificadores son condicionalmente independientes respecto a los subconjuntos K_i se obtiene la igualdad siguiente:

$$P(K_i|d) = P(K_i) \prod_{j=1}^{l} P(d_j|K_i). \quad (10)$$

En la ecuación anterior el término $P(K_i)$ representa la probabilidad a priori del subconjunto K_i de ocurrir, es decir la probabilidad de que un objeto cualquiera pertenezca al subconjunto K_i. Y los términos $P(d_j|K_i)$ codifican la probabilidad de que el clasificador D_j dé el resultado d_j cuando el objeto pertenece al subconjunto K_i.

El método de combinación entonces solo tendría que estimar todas las probabilidades $P(K_i|d)$ y $P(K_i)$; y dar como resultado una tupla con dichos valores como soporte para cada uno de los K_i. Este método se conoce como *combinación ingenua de Bayes* debido a que se asume independencia entre los clasificadores una vez conocida la clase a la que pertenece el objeto.

Para implementar este método es necesario estimar $P(K_i)$ y cada una de las $P(d_j|K_i)$ a partir de un conjunto de entrenamiento Z. Siendo N_i la cantidad de objetos de Z que pertenecen K_i, entonces $N_i/|Z|$ es un estimado de $P(K_i)$. Sea $cm_{i,j}^{k}$ el número de objetos en Z que pertenecen a K_i y que el clasificador D_k le asigna el subconjunto K_j, y siendo $d_j = K_m$ es posible estimar $P(d_j|K_i)$ a través de $cm_{i,m}^{j}/N_i$, esto es, la razón entre de los objetos que pertenecen a K_i y que D_j asigna a K_m y el total de objetos que pertenecen a K_i.

Es importante notar que si la estimación de algún $P(d_j|K_i)$ es igual a cero automáticamente el soporte de la combinación para el subconjunto K_i se hace cero. Existen propuestas para enfrentar este problema, entre ellas se encuentra la realizada en [34] que realiza la estimación de $P(d|K_i)$ a través de la fórmula $\left(\prod_{i=1}^{l}\left((cm_{i,m}^{j} + (1/k))/(N_i + 1)\right)\right)^{B}$, donde B es una constante, k es la cantidad de subconjuntos K_i, y se asume $d_j = K_m$.

El método tiene dos grandes debilidades, una teórica, que es que se asume independencia condicional entre los clasificadores, y una práctica, que surge de la necesidad de estimar ciertas probabilidades a partir de un conjunto de entrenamiento. Sin embargo, en la práctica el método ha demostrado tener una eficacia sorprendente ya que aunque asume condiciones que difícilmente se verifican en la práctica aun así los resultados son buenos con respecto a otros métodos de combinación, por ejemplo véase la comparación realizada en [1]. Además tiene la ventaja de ser simple y por consiguiente fácil de implementar.

3.1.3 BKS (Behavior Knowledge Space)

El BKS [35] [36] forma parte de un grupo de métodos de combinación llamados multinomiales, dichos métodos estiman la mejor salida de la combinación para cada una de todas las posibles tuplas d.

El espacio de conocimiento del comportamiento (BKS por sus siglas en inglés) no es más que un espacio l-dimensional donde cada dimensión se corresponde con la decisión de un clasificador. Cada punto en este espacio será denotado por $BKS(d)$ y tiene asociadas los siguientes parámetros, que serán calculados a partir de un conjunto de entrenamiento:

- $cant(d)$ que representa la cantidad total de objetos que registraron la salida d, o sea, este punto en el espacio BKS.
- $rep(d)$ que representa el subconjunto K_i que contiene a más objetos de los que registraron esta salida, es decir el K_i más representativo.
- $sub(d,\ K_i)$ que representa la cantidad total de objetos que pertenecen a K_i que registraron esta salida.

Nótese que por cada punto se asume que solo un K_i puede ser el más representativo, por lo que en caso de empate hay que emplear alguna técnica alternativa que puede variar según se desee. Lo que puede constituir un eslabón débil en la aplicación del método dado que la solución dependería del representante seleccionado por dicho método.

Una vez modelado el espacio BKS y obtenida la tupla $d = (d_1, \dots, d_l)$ a partir de las salidas de los clasificadores individuales. La combinación se lleva a cabo a través de la siguiente regla:

- El objeto se asigna al subconjunto K_i si se cumple:

$$cant(d) > 0 \wedge \frac{sub\big(d, rep(d)\big)}{cant(d)} \geq \mu \wedge rep(d) = K_i.$$

- Es decir, si se registró al menos una salida igual a d, y si dentro de todos los objetos que registraron la salida d la razón entre la cantidad de objetos que pertenecen al conjunto más representativo y el total es mayor que un umbral prefijado μ, entonces la salida de la combinación es el subconjunto más representativo de la salida d.
- La combinación se abstiene en otro caso.

Nótese en la ecuación anterior la presencia del umbral μ para controlar la confidencia en la decisión de esta regla.

En [36] son discutidas las principales propiedades del método, en particular las siguientes:

- Es un método de combinación óptimo cuando los clasificadores ofrecen solo –y nótese bien la palabra *solo*– una salida abstracta. Realmente el método es óptimo asintóticamente, es decir, según crezca el conjunto de entrenamiento más se acercará a la combinación óptima. Nótese que un método de combinación óptimo daría como resultado el K_i con mayor $P(K_i|d)$. Entonces como según el conjunto de entrenamiento se acerca al tamaño del universo la estimación de $P(K_i|d)$ se acerca también al valor real de $P(K_i|d)$ sucede que el BKS es un método asintóticamente óptimo pues selecciona al conjunto más representativo como resultado que sería entonces el de mayor $P(K_i|d)$.
- Se hace fácil estimar el mejor umbral μ tal que se obtenga un desempeño deseado. Es decir, una vez que se conocen los valores de probabilidades de sustitución, de abstención, de clasificación

correcta que serían ideales; automáticamente se puede ajustar μ para que la combinación tenga un desempeño parecido al deseado. Para más información consúltese [36].

- No asume independencia entre los clasificadores.

Las propiedades anteriormente expuestas evidencian las ventajas de este método de combinación. Sin embargo este presenta un problema bien serio desde el punto de vista práctico, y es el tamaño del espacio BKS, el cual crece exponencialmente según crece l pues contiene $|\Omega|^l$ elementos. En [36] se realizan algunas propuestas para afrontar este problema. Otros problemas según todos los autores ya citados sobre este enfoque son:

- Sufre el problema de muestra pequeña (*small sample size problem*). Para que este método funcione bien es necesario proveer conjuntos de entrenamiento suficientemente grandes.
- Presenta un alto grado de error en los puntos en los cuales la probabilidad del subconjunto más representativo no es muy alta.
- Muchas veces suele sobre-entrenarse.

Para enfrentar estos problemas es que precisamente se introducen el umbral y la posibilidad de abstención en la regla, aunque sigue sin ser suficientes. En [37] es propuesto un análisis del error de generalización del método, así como un modelo que relaciona el error al tamaño de la muestra en un punto del espacio BKS.

Según [38] es posible afirmar que la eficacia del método de combinación BKS, al ser un método de combinación que requiere entrenamiento, aumenta con relación a otros métodos de combinación que no necesitan entrenamiento según:

- Aumente el tamaño del conjunto de entrenamiento.
- Disminuya la cantidad de clasificadores.
- Exista un desbalance en la eficacia de cada uno de los clasificadores, o sea, en la combinación pueden haber clasificadores individuales muy eficaces y otros no tan eficaces.

3.1.4 Método de Wernecke

El método propuesto por Wernecke [39] es otro método multinomial muy similar al BKS, lo que este presta especial atención al problema del sobre-entrenamiento. El método considera intervalos de confianza del 95% sobre la cantidad de objetos que se registran en cada punto del espacio por cada una de las clases –recuérdese que un punto es una combinación posible de las salidas del clasificador–, si existe alguna intersección entre los intervalos, entonces la clase con más prevalencia no es considerada, en cambio se toma la decisión del clasificador "menos malo".

El clasificador "menos malo" es el clasificador D_j que registra el mínimo de l estimaciones de la probabilidad $P(error \wedge D_j(x) = d_j)$. En palabras, se toma el clasificador D_j que haya clasificado menos objetos dentro de d_j de manera incorrecta. Dicho clasificador se infiere a partir del conjunto de entrenamiento.

En estadística, los intervalos de confianza son utilizados para garantizar que con cierta probabilidad el valor de cierto argumento es contenido en su interior. En el caso del método de Wernecke por cada punto en el espacio BKS se crean intervalos que expresen la confidencia de los valores $sub(d,\ K_i)$.

Para simplificar la notación hagamos $N_i = sub(d, K_i)$ y a $N = cant(d)$. Para calcular los intervalos de confianza al 95% se asume que cada valor N_i se distribuye a partir de la ley binomial y se utiliza la aproximación a la distribución normal o la desigualdad de Chebyshev para calcular los intervalos. Por ejemplo, utilizando la aproximación a la distribución normal el intervalo de confianza del 95% para el conjunto K_i quedaría expresado:

$$IC(K_i, 95) = [N_i - 1.96\sqrt{\frac{N_i(N - N_i)}{N}} + \frac{1}{2}, N_i + 1.96\sqrt{\frac{N_i(N - N_i)}{N}} - \frac{1}{2}].$$

El método funciona entonces de la siguiente manera:

1. Si existe un empate en cuanto al máximo N_i entonces no es necesario calcular los intervalos de confianza y se devuelve el resultado del clasificador "menos malo".
2. En otro caso, es decir, existe una clase que predomina, se deben calcular los intervalos de confianza y si el intervalo correspondiente a la clase predominante no se intersecta con ningún otro entonces se da como resultado dicha clase.
3. En cambio, si existe una intersección, entonces el resultado es el del clasificador "menos malo".

Cuando el conjunto de entrenamiento no sea lo suficientemente grande Wernecke [39] propone utilizar la inecuación de Chebyshev. Es importante notar que según se amplían los intervalos, se está dejando de un lado el enfoque multinomial para pasar a un enfoque de combinación a través del clasificador "menos malo", cuando esto ocurre se recomienda usar otros porcientos para los valores de confianza, por ejemplo 70% [1].

El método de Wernecke presenta todos los problemas que presentan los métodos multinomiales (BKS), sin embargo realiza un mayor esfuerzo en el aspecto de evitar el sobre-entrenamiento.

3.2 Combinación de salidas de tipo rango

En [40] Ho hace notar que las salidas de tipo rango contienen más información que las de tipo abstracto, sobre todo cuando el número de subconjuntos K_i aumenta, pues en estos casos las segundas y terceras opciones aportan información que no debe ser pasada por alto. Por otra parte, cuando las salidas de los clasificadores son del tipo de medida, puede ocurrir que la diferencia de escala, o incompatibilidades entre las medidas de los clasificadores haga muy complejo sino imposible su combinación.

En la Biometría por ejemplo, es usual que el número de clases sea muy grande, además casi siempre es deseable que el clasificador dé varios candidatos. Por ejemplo, si se detecta una huella en la escena de un crimen –que por otra parte puede que no sea muy buena– es deseable buscar en una base de datos de personas –que suele ser muy grande– los mejores candidatos.

Ho propone dos enfoques para la combinación de clasificadores con salida de tipo rango: la reducción del conjunto de clases disponibles y el reordenamiento de los subconjuntos K_i. En el primero, el objetivo es extraer un subconjunto de Ω tan pequeño como sea posible y que todavía contenga el K_i que contenga el objeto en cuestión. En el segundo, como su nombre lo indica el objetivo es dar un nuevo ordenamiento de Ω que exprese un consenso entre los clasificadores individuales. A continuación se expondrán dichos métodos con cierto detalle.

3.2.1 Métodos de reducción del conjunto de las clases

Ho [40] propone dos métodos para la reducción del conjunto de clases, y ambos requieren una fase de entrenamiento. El primero se denomina *método de intersección de grandes vecindades*, pues computa la intersección de conjuntos que contienen los K_i obtenidos de la salida de cada clasificador, y se espera que estos conjuntos tengan un tamaño grande. Esto se debe a que el conjunto es determinado a partir del rango más bajo que se le dio a una clase correcta en un conjunto de entrenamiento. Es decir, que un clasificador dé un posicionamiento lejano a la clase correcta para un objeto en el conjunto de entrenamiento, implicará que los conjuntos que este aporte al método de combinación sean grandes. Por otra parte, clasificadores eficaces sobre todos los objetos en el conjunto de entrenamiento siempre darán conjuntos pequeños.

La Tabla 6 muestra el funcionamiento del método para un caso con cuatro clasificadores, cuatro clases y cinco objetos en el conjunto de entrenamiento. Primeramente se muestran los cinco objetos en el conjunto de entrenamiento $(O_1, \dots, O_5)$ así como la clase a la que estos pertenecen, seguidos de los resultados obtenidos por los cuatro clasificadores individuales. Los resultados de los clasificadores con cada uno de estos objetos quedan entonces en columnas. Por ejemplo, el tercer clasificador (D_3) da el peor rango a una clase correcta con el quinto objeto, le da rango 2, esto es, se tiene que $O_5 \in K_1$ y sin embargo $D_3(O_5)$ pone a K_1 en el segundo lugar del ordenamiento. Entonces cuando D_3 clasifica un nuevo objeto no se tienen en cuenta las clases con un rango más bajo que este, nótese que ya en fase de

clasificación con el sexto objeto el conjunto resultante de dicho clasificador solo contiene 2 subconjuntos K_i. Es fácil notar que un clasificador es redundante cuando el tamaño del conjunto seleccionado para él es igual al número de K_i presentes en el problema, en el ejemplo anterior, son redundantes el primero y el segundo.

Tabla 6. Ejemplo de funcionamiento del método de intersección de grandes vecindades.

Fase de entrenamiento.																	
Objetos	K_i correcto	Clasificador D_1				Clasificador D_2				Clasificador D_3				Clasificador D_4			
		K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4
O_1	K_1	1	2	3	4	2	1	4	3	1	3	2	4	1	2	3	4
O_2	K_4	3	1	4	2	1	2	3	4	3	2	4	1	1	2	4	3
O_3	K_2	4	1	2	3	2	3	1	4	2	1	3	4	1	2	3	4
O_4	K_3	4	1	2	3	3	4	1	2	2	3	1	4	4	2	3	1
O_5	K_1	4	2	3	1	1	2	3	4	2	1	3	4	1	4	2	3
Umbrales		4				4				2				3			
Fase de clasificación.																	
O_6		1	2	3	4	4	3	1	2	1	3	4	2	2	1	3	4
Conjuntos seleccionados		$\{K_1, K_2, K_3, K_4\}$				$\{K_1, K_2, K_3, K_4\}$				$\{K_1, K_4\}$				$\{K_1, K_2, K_3\}$			
Conjunto resultante de la intersección		$\{K_1\}$															

El segundo método se denomina *método de unión de vecindades pequeñas*, pues se calcula la unión de conjuntos seleccionados de la salida de cada clasificador, el tamaño de cada conjunto se define a partir del conjunto de entrenamiento de la siguiente manera: por cada clasificador D_i, cada vez que este asigne el mejor rango a la clase correcta, se almacenará en una lista. Entonces el mayor de estos rangos por cada clasificador determinará el tamaño del conjunto de ese clasificador.

En la Tabla 7 se muestra el funcionamiento del segundo método en los mismos casos que la Tabla 6. Por cada clasificador se almacena un número por cada objeto en el conjunto de entrenamiento: el rango que le asignó a la clase correcta si fue el mejor clasificador, o empató con los mejores, y cero en otro caso. Por cada clasificador se toma el máximo de estos números para definir el tamaño del conjunto de clases a sacar de la salida de los clasificadores, para los casos mostrados siempre es uno. Es importante notar que cuando para un clasificador, el máximo de sus números almacenados es cero –siempre hubo un clasificador mejor que él durante el entrenamiento–, se considera redundante y no debe ser incluido en la unión final.

En [40] se explica que como el método de intersección se basa en el caso peor, resulta beneficioso usarlo cuando los clasificadores tienen un desempeño aceptable considerando el caso peor, pues en caso contrario el tamaño de los conjuntos será demasiado grande. Sin embargo, se conoce que este no es el caso para un conjunto de clasificadores especializados solo en un conjunto de las características de los objetos, o simplemente especializados en cierta parte del espacio de representación pues estos clasificadores tienden a clasificar muy bien a unos objetos y muy mal a otros. De esta forma el método de la unión es preferible cuando los clasificadores están especializados en ciertos tipos de entrada.

Tabla 7. Ejemplo del funcionamiento del método de unión de pequeñas vecindades.

Fase de entrenamiento.																	
Objetos	K_i correcto	Clasificador D_1				Clasificador D_2				Clasificador D_3				Clasificador D_4			
		K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4	K_1	K_2	K_3	K_4
O_1	K_1	1	2	3	4	2	1	4	3	1	3	2	4	1	2	3	4
O_2	K_4	3	1	4	2	1	2	3	4	3	2	4	1	1	2	4	3

O_3	K_2	4	1	2	3	2	3	1	4	2	1	3	4	1	2	3	4
O_4	K_3	4	1	2	3	3	4	1	2	2	3	1	4	4	2	3	1
O_5	K_1	4	2	3	1	1	2	3	4	2	1	3	4	1	4	2	3
		Mejores rangos por clasificador.															
O_1	K_1	1				0				1				1			
O_2	K_4	0				0				1				0			
O_3	K_2	1				0				1				0			
O_4	K_3	0				1				1				0			
O_5	K_1	0				1				0				1			
Umbrales		1				1				1				1			
Fase de clasificación.																	
O_6		1	2	3	4	4	3	1	2	1	3	4	2	2	1	3	4
Conjuntos seleccionados		$\{K_1\}$				$\{K_3\}$				$\{K_1\}$				$\{K_2\}$			
Conjunto resultante de la intersección		$\{K_1, K_2, K_3\}$															

Queda claro que un subconjunto de las clases no tiene mucha aplicabilidad desde el punto de vista práctico, pues aunque la clase correcta tenga gran probabilidad de estar contenida aún no se conoce cuál es, en [40] no se plantean esto como un problema pendiente, sin embargo, pensamos que en algunos casos sería conveniente volver a establecer un ordenamiento entre los elementos del conjunto final.

3.2.2 Métodos de reordenamiento (re-ranking) de las clases

Los métodos de reordenamiento de las clases intentan mejorar la posición en la que se encuentra la clase correcta. Ho propone en [40] tres soluciones de este tipo y a continuación se exponen las dos más conocidas.

Método del rango mayor

Cada objeto recibe l posicionamientos no necesariamente diferentes, pues se tienen l clasificadores. El método del rango mayor, en el reordenamiento resultante, coloca cada clase en la mejor de las l posiciones asignadas por cada clasificador, los empates lamentablemente deben ser resueltos de manera arbitraria. Nótese que aunque se aplique una técnica como darle prioridad de la clase que más veces fue posicionada en la respectiva mejor posición, aún pueden seguir ocurriendo empates. Este método aprovecha la fortaleza de cada clasificador, y esto puede ser beneficioso pero también indeseable. Supongamos que cuando se clasifica un objeto un clasificador pone la clase correcta en una posición ventajosa, entonces en el ordenamiento final –si no hay muchos empates– la clase correcta seguirá estando en una posición ventajosa, pero por otra parte si un clasificador pone una clase incorrecta en una posición ventajosa, entonces dicha clase en el resultado final pudiera seguir en dicha posición. Este método da mejores resultados cuando el número de clasificadores es pequeño con respecto al número de clases, en caso contrario tiende a ocurrir demasiados empates, lo que hace que el método pierda sentido ya que como se vio los empates se resuelven de manera aleatoria.

Método del conteo de Borda

El método del conteo de Borda, es un método de decisión en elecciones donde los votantes se expresan a través de un ordenamiento de sus opciones, fue desarrollado independientemente por varios autores pero debe su nombre al matemático francés Jean-Charles de Borda que le dio su forma definitiva en 1770. Sea $b_{i,j}$ el número de clases que se encuentran en una posición peor que la clase K_i en el resultado obtenido del clasificador D_j al clasificar un objeto. Sea $B_i = \sum_{j=1}^{l} b_{i,j}$, es decir, B_i expresa

cuán bien está ubicado K_i a través de todos los clasificadores. Entonces el ordenamiento final viene determinado por el ordenamiento de forma descendente de todos los B_j.

Por ejemplo consideremos el siguiente ejemplo (Tabla 8) con los resultados de 6 clasificadores para 6 clases. En cada celda de la tabla se muestra la posición otorgada por el clasificador que representa la respectiva columna a la clase que representa la respectiva fila, además entre paréntesis se coloca el valor $b_{i,j}$ correspondiente, excepto para la última columna que muestra los valores B_i correspondientes. En el ejemplo mostrado el ordenamiento que resulta a partir del conteo de Borda es $(K_4, K_5, K_5, K_1, K_2, K_3)$.

Tabla 8. Ejemplo del funcionamiento del conteo de Borda.

	D_1	D_2	D_3	D_4	D_5	D_6	B_i
K_1	2 ($b_{1,1}=4$)	1 ($b_{1,2}=5$)	6 ($b_{1,3}=0$)	4 ($b_{1,4}=2$)	3 ($b_{1,5}=3$)	4 ($b_{1,6}=2$)	$B_1=16$
K_2	5 ($b_{2,1}=1$)	5 ($b_{2,2}=1$)	4 ($b_{2,3}=2$)	5 ($b_{2,4}=1$)	5 ($b_{2,5}=1$)	3 ($b_{2,6}=3$)	$B_2=9$
K_3	6 ($b_{3,1}=0$)	6 ($b_{3,2}=0$)	3 ($b_{3,3}=3$)	6 ($b_{3,4}=0$)	6 ($b_{3,5}=0$)	2 ($b_{3,6}=4$)	$B_3=7$
K_4	1 ($b_{4,1}=5$)	2 ($b_{4,2}=4$)	5 ($b_{4,3}=1$)	3 ($b_{4,4}=3$)	4 ($b_{4,5}=2$)	1 ($b_{4,6}=5$)	$B_4=20$
K_5	4 ($b_{5,1}=2$)	4 ($b_{5,2}=2$)	1 ($b_{5,3}=5$)	1 ($b_{5,4}=5$)	2 ($b_{5,5}=4$)	5 ($b_{5,6}=1$)	$B_5=19$
K_6	3 ($b_{6,1}=3$)	3 ($b_{6,2}=3$)	2 ($b_{6,3}=4$)	2 ($b_{6,4}=4$)	1 ($b_{6,5}=5$)	6 ($b_{6,6}=0$)	$B_6=19$

El conteo de Borda tiene en cuenta el consenso entre los clasificadores, pues si todos los clasificadores posicionan una clase cerca de la primera posición entonces su número de Borda –el B_j correspondiente– será alto por lo que estará entre las primeras posiciones del ordenamiento final. En cambio, si un solo clasificador es el que pone a cierta clase en una posición privilegiada esta no tendrá por qué seguir en una posición cercana al primer lugar en el reordenamiento final. Nótese que aún pueden suceder empates, según Ho propuestas para enfrentar este problema pueden ser encontradas en [41].

Este método trata a todos los clasificadores por igual, lo que puede no ser deseable y una solución pudiera ser agregar pesos a cada $b_{i,j}$ que expresen la calidad del clasificador D_i.

3.3 Combinación de salidas de tipo medida

Cuando la salida de un clasificador es del tipo medida, este devuelve una tupla al clasificar un objeto la cual contiene los grados de soportes que dicho clasificador da a la hipótesis de que el objeto pertenezca a cada una de las clases. Estos grados de soporte pueden ser interpretados de distintas maneras, por ejemplo: como estimaciones de la probabilidad a posteriori del objeto perteneciendo a cada clase, o sea $P(K_i|x)$, como la similitud con los prototipos de cada una de las clases, como grados de pertenencia difusa, etc. Es importante notar que las salidas de medida no pueden ser interpretadas de manera arbitraria, pues eso depende del clasificador en cuestión, por ejemplo, las salidas de los clasificadores de discriminante lineal y cuadrático no pueden ser interpretadas directamente como las probabilidades a posteriori de cada clase [1], tampoco es correcto interpretar los soportes como distancias o similitudes a un prototipo de la clase sin ningún fundamento; por otra parte, a veces se plantean teorías acerca de las propiedades que cumplen las salidas de ciertos clasificadores realizando asunciones que no se verifican en la realidad, por ejemplo se dice que un clasificador da como resultado la estimación de cada una de las probabilidades a posteriori $P(K_i|x)$ si se verifica alguna propiedad –por ejemplo, que la función de optimización interna es capaz de obtener un óptimo global–, sin embargo esta última no se verifica en la práctica. En [1] se pueden encontrar más detalles sobre esta cuestión. No obstante, existen técnicas que permiten la conversión de las salidas de grados de soportes de algunos clasificadores, que pueden tener una interpretación específica, varios ejemplos pueden ser encontrados en [1].

En este trabajo, al igual que en [1], los métodos de combinación de salidas de tipo medida serán divididos en dos tipos:

- Los que para dar el soporte final a una clase solo tienen en cuenta los soportes de los clasificadores individuales a esa misma clase.
- Los que para dar el resultado de la combinación observan la matriz de perfil de decisión completa.

A los primeros se les llama *funciones de combinación conscientes de la clase* (*class-conscious*), a los segundos *funciones de combinación indiferentes a la clase* (*class-indifferent*).

A continuación se expondrán un conjunto de funciones de combinación conscientes de la clase y después un conjunto de funciones de combinación indiferentes a la clase.

3.3.1 Funciones de combinación conscientes de la clase

Las funciones de combinación conscientes de la clase siempre dan como resultado una salida de tipo medida pues, analizan los soportes de todos los clasificadores individuales por cada una de las clases. A continuación se exponen cuatro métodos de combinación de este tipo, los cuales se encuentran entre los más usados.

Combinación a través de funciones

Los métodos de combinación a través de funciones utilizan una función para combinar los soportes que dan todos los clasificadores a cada clase. Si denotamos por f a dicha función, entonces el grado de soporte que otorgará la combinación a la clase K_j –el cual denotaremos por t_j– sería el siguiente:

$$t_j = f(DP(1,j), \dots, DP(l,j)). \tag{11}$$

En la ecuación anterior DP representa la matriz de perfil de decisión. Los soportes t_j forman una tupla que es el resultado de la combinación de los clasificadores individuales.

Entre las funciones más utilizadas para este propósito se encuentran las siguientes:

- Promedio $t_j = \frac{1}{l}\sum_{i=1}^{l} DP(i,j)$.
- Máximo $t_j = \max_i\{DP(i,j)\}$.
- Mínimo $t_j = \min_i\{DP(i,j)\}$.
- Producto $t_j = \prod_{i=1}^{l} DP(i,j)$.
- Promedio generalizado $t_j = \left(\frac{1}{l}\sum_{i=1}^{l} DP(i,j)^{\alpha}\right)^{1/\alpha}$ donde α es un parámetro.

La función de mínimo es la opción más pesimista pues cuando es usada se desea que el soporte final del esquema no sea mayor que cualquier soporte dado por los clasificadores individuales. En el otro extremo, la función de máximo es la opción más optimista ya que le es suficiente el mayor soporte de los clasificadores individuales. La función producto tiene el problema de que un soporte igual a cero de algún clasificador invalida los soportes de los restantes clasificadores. El promedio generalizado tiene un argumento extra α, que debe ser ajustado. Dicho argumento puede ser interpretado como un grado de optimismo ya que cuando este tiende al infinito negativo el promedio generalizado y el mínimo se hacen funciones equivalentes, y cuando este tiende al infinito positivo el promedio generalizado es equivalente a la función máximo. El parámetro α puede ser fijado arbitrariamente, sin embargo no existe un consenso acerca de cuál es el valor adecuado a asignarle pues el desempeño final depende del problema en concreto y de los clasificadores individuales que se utilizan [1].

Nótese que no es necesario que los soportes de los clasificadores sumen uno, de hecho, aunque el único requerimiento que imponen estas funciones es que sus parámetros sean números reales, se debe prestar especial atención a lo que se va a combinar, pues se asume que los soportes son medidas en una misma unidad, sino ¿qué sentido tendría multiplicar un grado de similitud por una probabilidad, o que utilizando la función máximo el resultado final mezcle distintos tipos de soporte? En general, las ventajas que poseen estos métodos son su sencillez y facilidad de implementación.

Promedio ponderado ordenado

Este método utiliza l coeficientes, los cuales serán asociados a los clasificadores individuales, aunque no de una manera predeterminada. Primeramente los soportes de todos los clasificadores a una clase son ordenados de manera descendiente y luego se realiza una suma ponderada con los l coeficientes, es decir, los coeficientes se asocian a una posición del ordenamiento en lugar de a un clasificador.

Sea $b = (b_1, b_2, \dots, b_l)$ una tupla de coeficientes tal que $\sum_{k=1}^{l} b_k = 1$ y $0 \leq b_k \leq 1$, entonces el grado de soporte que da la combinación a la clase K_j es calculado según la siguiente fórmula:

$$t_j = \sum_{k=1}^{l} (b_k * DP(i_k, j)). \quad (12)$$

En la ecuación anterior los índices $i_1, i_2, \dots, i_l$ representan una permutación de los índices $1, 2, \dots, l$ tal que se cumple $DP(i_1, j) \geq DP(i_2, j) \geq \cdots \geq DP(i_l, j)$.

Este método de combinación permite modelar distintos modos de combinación. Por ejemplo, en muchas competencias deportivas, antes de hallar el promedio, la mejor y la peor votación de los jueces son eliminadas con el objetivo de mitigar una posible parcialidad, esto es posible de lograr con el promedio ponderado ordenado utilizando la siguiente tuplas de coeficientes $(0, \frac{1}{l-2}, \dots, \frac{1}{l-2}, 0)$. Para modelar las funciones de combinación mínimo y máximo se usan las tuplas de coeficientes $(0, 0, \dots, 1)$ y $(1, 0, \dots, 0)$ respectivamente.

El promedio ponderado ordenado también asume que los soportes de los clasificadores se encuentran en una misma unidad de medida.

Promedio ponderado

En el promedio ponderado se utilizan varios coeficientes –pesos– que serán asociados a las decisiones de los clasificadores –ahora sí se asocian los pesos a los clasificadores– con el objetivo de tener más en cuenta las decisiones de los mejores clasificadores y menos en cuenta las decisiones de los clasificadores menos precisos. Existen distintos enfoques:

- Utilizar l coeficientes $b_1, b_2, \dots, b_l$, uno por cada clasificador. La función de combinación quedaría $t_j = \sum_{i=1}^{l} b_i * DP(i, j)$. La opción más usada es tomar los b_i a partir de la estimación de la eficacia de cada clasificador.
- Utilizar $k * l$ coeficientes, uno por cada clasificador y cada clase. Se asume que ciertos clasificadores serán muy precisos con ciertas clases y a su vez muy imprecisos con otras. La función quedaría $t_j = \sum_{i=1}^{l} b_{i,j} * DP(i, j)$.

En la literatura se han utilizado diversos métodos para estimar los coeficientes, ejemplos pueden ser encontrados en [42], [43], [44], [45]. Al igual que en los métodos anteriores se asume que los soportes que otorgan los clasificadores individuales son conceptualmente homogéneos, pues ¿qué sentido tendría promediar probabilidades, distancias, grados de similitud?

Integrales difusas

El concepto de medida ha sido largamente usado en la Matemática. Dado un conjunto A una medida se define como una función μ sobre una σ-álgebra $\sum$ –colección de subconjuntos de A cerrada sobre el complemento, la unión y la intersección– definida sobre A, e imagen en los reales no negativos. Además la función μ debe cumplir las siguientes propiedades:

- $\mu(\emptyset) = 0$.
- Aditividad. $\forall B_i \in \sum, B_j \in \sum [B_i \cap B_j = \emptyset \rightarrow \mu(B_i \cup B_j) = \mu(B_i) + \mu(B_j)]$.

Como ejemplos de σ-álgebra se puede mencionar el conjunto potencia de cualquier conjunto, o el álgebra de Borel sobre cualquier espacio topológico. Como ejemplos de medidas se pueden citar la

medida de Lebesgue, o las funciones de probabilidad. Para más información el lector puede consultar [46]

Intentando dar otro enfoque en la generalización del concepto de medida, en la segunda mitad de siglo pasado se definen por primera vez los conceptos de medida difusa e integral difusa. Aunque el término de medida difusa ha sido aceptado por la comunidad científica, también ha causado confusión ya que la palabra difusa no significa que la medida se aplique a conjuntos difusos ni tampoco que la medida sea difusa en el sentido de la teoría de conjuntos difusos. Una medida difusa es una función monótona no negativa de valores definidos en conjuntos clásicos. En algunos libros el término de medida difusa es sustituido por términos que traen menos confusión, como medidas monótonas, medidas no aditivas, medidas generalizadas, entre otros.

Las medidas difusas son creadas básicamente por las limitaciones cada vez más aceptadas de la teoría clásica, en particular las limitaciones que trae consigo la exigencia de la propiedad de la aditividad. Vale la pena aclarar, dichas limitaciones no son limitaciones en sí, sino limitaciones de su aplicación en ciertos contextos, o sea la utilización de *medidas no aditivas* –nótese que existe una contradicción en el término ya que las medidas son aditivas por definición– en determinados contextos ofrece enfoques más realistas a una gama de problemas.

Entonces una función μ definida en una σ-álgebra $\sum$ de un cierto conjunto A y que tiene como imagen el conjunto de los números reales no negativos es una medida difusa si esta cumple las siguientes propiedades:

- $\mu(\emptyset) = 0$.
- Monotonía. $\forall B_i \in \sum, B_j \in \sum [B_i \subset B_j \rightarrow \mu(B_i) \leq \mu(B_j)]$.

Es importante notar que existe en la literatura una gran cantidad de definiciones de medidas difusas las cuales varían en las propiedades que son exigidas.

En 1974 Sugeno [47] introduce el concepto de lambda-medida, que no es más que un caso particular de medida difusa. Sean un conjunto A y una σ-álgebra $\sum$ definida sobre A, sea además $\lambda \in (-1, \infty)$, entonces una lambda-medida –λ-medida– es una función μ_λ definida sobre $\sum$ y que tiene como imagen el intervalo real [0,1] que es una medida difusa y satisface:

- $\mu_\lambda(A) = 1$.
- $\forall B_i \in \sum, B_j \in \sum [B_i \cap B_j = \emptyset \rightarrow \mu_\lambda(B_i \cup B_j) = \mu_\lambda(B_i) + \mu_\lambda(B_j) + \lambda * \mu_\lambda(B_i) * \mu_\lambda(B_j)]$.

Para un conjunto A, si se considera su conjunto potencia como σ-álgebra correspondiente es posible construir una λ-medida de la siguiente forma:

- Se fijan los valores de μ_λ para todo los subconjuntos unitarios de A.
- Se obtiene el valor de λ a partir de la ecuación

$$1 + (\lambda * \mu_\lambda(A)) = \prod_{a_i \in A}(1 + \lambda * \mu_\lambda(\{a_i\})), \qquad (13)$$

- Nótese que una vez definidos los valores de la medida para los conjuntos unitarios y hallado el valor de λ se tiene completamente definida la λ-medida correspondiente, pues la λ-medida de cualquier subconjunto se puede calcular recursivamente.

La función μ_λ restringida a los subconjuntos unitarios del conjunto A define los valores que serán llamados *valores de densidad difusa*, y denotados por g_i ($\forall a_i \in A [g_i = \mu_\lambda(\{a_i\})]$). Se verifica que para un conjunto de valores de densidad difusa existe un único valor λ tal que $\lambda \in (-1, \infty)$ y $\lambda \neq 0$ que verifique la ecuación (13).

La noción de las integrales difusas surge a la par de las medidas difusas. Entre las integrales difusas más conocidas se encuentran la integral de Choquet y la integral de Sugeno. Sean $A = \{a_1, \dots, a_n\}$ un conjunto finito, $f: A \rightarrow [0, \infty]$ una función tal que $f(a_1) \leq \cdots \leq f(a_n)$, y $A_i = \{a_i, \dots, a_n\}$. Entonces la integral de Sugeno de f a partir de una medida difusa μ se define:

$$\int f \, d\mu = \max_{1 \leq i \leq n}(\min(f(a_i), \mu(A_i))). \qquad (14)$$

Por otra parte, para definir la integral de Choquet se define a $A_{n+1} = \emptyset$, entonces la integral quedaría

$$\int f \, d\mu = \sum_{i=1}^{n} (f(a_i) * (\mu(A_i) - \mu(A_{i+1}))). \quad (15)$$

En el contexto de la combinación de clasificadores el conjunto A sería el conjunto de clasificadores que se desean combinar, los valores de densidad difusa (los g_i) deben ser interpretados como una media de la competencia eficacia del clasificador individual D_i, nótese que para clases distintas se pueden tener distintos valores de densidad difusa (o sea distinta competencia). La función $f: A \to [0, \infty]$ no sería más que el grado de soporte que da un clasificador individual a la hipótesis de que un objeto pertenece a la clase K_j, o sea $f(D_i) = DP(i,j)$ para la K_j en cuestión. De esta forma, una vez definidos los valores de densidad difusa correspondientes, y hallada λ, el soporte de la combinación a cada clase se computa integrando la función definida a través de los soportes individuales de los clasificadores para la clase en cuestión, con respecto a la λ-medida correspondiente.

Por ejemplo, si se tienen tres clasificadores D_1, D_2, D_3, y se le asignan los siguientes valores de densidad difusa $g_1 = 0.4$, $g_2 = 0.4$ y $g_3 = 0.5$, luego sustituyendo en la Ecuación 13 se obtiene que $\lambda = -0.584$, es decir la λ-medida obtenida es la siguiente:

- $\mu_\lambda(\{D_1\}) = 0.4$
- $\mu_\lambda(\{D_2\}) = 0.4$
- $\mu_\lambda(\{D_3\}) = 0.5$
- $\mu_\lambda(\{D_1, D_2\}) = \mu_\lambda(\{D_1\}) + \mu_\lambda(\{D_2\}) + \lambda * \mu_\lambda(\{D_1\}) * \mu_\lambda(\{D_2\}) = 0.706$
- $\mu_\lambda(\{D_1, D_3\}) = \mu_\lambda(\{D_1\}) + \mu_\lambda(\{D_3\}) + \lambda * \mu_\lambda(\{D_1\}) * \mu_\lambda(\{D_3\}) = 0.78$
- $\mu_\lambda(\{D_2, D_3\}) = \mu_\lambda(\{D_2\}) + \mu_\lambda(\{D_3\}) + \lambda * \mu_\lambda(\{D_2\}) * \mu_\lambda(\{D_3\}) = 0.78$
- $\mu_\lambda(\{D_1, D_2, D_3\}) = 1$

Ahora supóngase que el soporte de los clasificadores a la hipótesis de que un objeto pertenece a una clase K_j es $DP(1,j) = 0.3$, $DP(2,j) = 0.2$ y $DP(3,j) = 0.7$. Primeramente utilizaremos una permutación de los índices de forma que se verifique que $f(D_{i_1}) \leq \cdots \leq f(D_{i_3})$, esto es $i_1 = 2, i_2 = 1, i_3 = 3$. Si se utiliza la integral de Sugeno (14) para calcular el soporte de la combinación a la clase K_j se tiene:

$$\max\{\min\{f(D_{i_1}), \mu(\{A_{i_1}\})\}, \min\{f(D_{i_1}), \mu(\{A_{i_1}\})\}, \min\{f(D_{i_1}), \mu(\{A_{i_1}\})\}\}$$
$$\max\{\min\{0.2,1\}, \min\{0.3,0.78\}, \min\{0.7,0.5\}\}$$
$$\max\{0.2,0.3,0.5\}$$

Es decir, el soporte que da la combinación a la clase K_j es 0.5. La combinación de clasificadores se pudo haber hecho utilizando la integral de Choquet. En [48] Kuncheva analiza los resultados de varios experimentos donde compara la eficacia de las funciones difusas de combinación, entre las que se encuentra la integral de Sugeno, y las funciones no difusas de combinación. En dichos experimentos se obtiene una superioridad por parte de las primeras y en particular la integral de Sugeno obtiene buenos resultados. Kuncheva señala la importancia de una estimación lo más precisa posible de los parámetros adicionales de la función de combinación y en el caso particular de las integrales difusas, de los valores de densidad difusa.

El método de combinación a través de integrales difusas asume que todos los clasificadores dan una salida del mismo tipo, pues nótese que se consideran los soportes de todos los clasificadores a una clase dada como una misma función, por lo que no tendría sentido que esa función devolviera para el primer clasificador una estimación de la probabilidad a posteriori de la clase una vez conocido el objeto y para el segundo la similitud del objeto con el prototipo correspondiente. Por otra parte es importante que los valores de densidad difusa expresen la confiabilidad tenida en el soporte del clasificador en cuestión.

3.3.2 Funciones de combinación indiferentes a la clase

Las funciones de combinación indiferentes a la clase no tienen que necesariamente dar una salida de tipo medida ya que es posible utilizar las salidas de los clasificadores como un nuevo espacio de representación y utilizar un nuevo clasificador –posiblemente uno de salidas de tipo abstracto– para dar la salida final de la combinación.

La práctica de utilizar las matrices de perfil de decisión como espacio intermedio de representación para nuevos clasificadores ha sido bastante utilizada, por ejemplo en [49] es utilizada una red neuronal. Creemos que es importante tener en cuenta que las distribuciones de las clases en dicho espacio pueden llegar a ser funciones muy complejas, por lo que podría ocurrir que clasificadores que asuman características acerca de dichas funciones –por ejemplo, que son distribuciones normales– fallen. Si se va a utilizar este enfoque recomendamos analizar cómo se distribuyen las clases en este espacio para el problema y los clasificadores individuales en cuestión.

Plantillas de decisión

La idea fundamental en el método de plantillas de decisión [50] es, por cada clase K_i, registrar una matriz de perfil de decisión, la cual se llamará plantilla de decisión, que se vincule más con todas las matrices de perfil de decisión obtenidas por los clasificadores individuales para objetos que pertenezcan a K_i.

En la fase de entrenamiento la plantilla de decisión para la clase K_i es computada promediando las matrices de perfil de decisión obtenidas para los objetos en el conjunto de entrenamiento que pertenecen a K_i. En la fase de clasificación el soporte a la clase K_i es obtenido a partir una medida de similitud entre la matriz de perfil de decisión actual y la plantilla de decisión asociada a K_i.

Las matrices de perfil de decisión y de plantilla de decisión pueden ser vistas como tuplas de $l * k$ componentes, y puede ser usada como función de similitud la inversa de cualquier función de distancia, por ejemplo la euclidiana, la Minkowski, o cualquier otra. Nótese que el método de plantillas de decisión no es más que la aplicación del clasificador que utiliza el promedio más cercano al espacio intermedio de las salidas de los clasificadores.

3.4 Selección de clasificadores

La selección de clasificadores es un enfoque distinto a los que se han estudiado hasta el momento, pues en lugar de tener en cuenta a todos los clasificadores para dar una salida final, se utiliza una componente normalmente conocida como oráculo que se encarga de seleccionar cuál o cuáles son los mejores clasificadores –los clasificadores más competentes o idóneos– para asignar un subconjunto K_i a un objeto en cuestión. En este sentido, nótese que para referirnos al clasificador idóneo para realizar la clasificación utilizaremos los términos competente y competencia que Kuncheva introduce en [1], en lugar de eficaz y eficacia ya que en un momento dado un clasificador que no es eficaz pudiera ser el más idóneo para realizar la clasificación, ya que un clasificador pudiera ser eficaz de manera general y fallar con los objetos que pertenecen a una pequeña región del espacio de representación. Este enfoque encaja perfectamente en nuestro modelo de función de combinación de salidas de los clasificadores ya que estas funciones reciben como parámetros el objeto a clasificar y las salidas de todos los clasificadores, es decir, toda la información necesaria para un oráculo.

Según Kuncheva [1] la idea de la selección de clasificadores se remonta al año 1978 cuando en el trabajo [51] se propone realizar una selección entre dos clasificadores, un k vecino más cercano y un discriminante lineal. El primero se utilizaba con los objetos que pertenecían a cierta zona del espacio de representación donde existía una mezcla entre los objetos de distintas clases, el segundo se utilizaba en la zona restante.

Un ejemplo típico muy utilizado por los autores en la literatura para demostrar la validez de la selección de clasificadores en ciertos contextos es el mostrado a continuación. La Figura 2 –tomada de [1]– muestra un espacio de representación con objetos de dos clases, los puntos claros representan una

clase K_1 y los más oscuros otra clase K_2. Si son considerados tres clasificadores: D_1 que siempre da como resultado la clase K_1, D_2 que siempre da como resultado la clase K_2 y D_3 que discrimina entre las clases según la línea discontinua mostrada en la figura. Cada uno de estos clasificadores tiene una eficacia aproximada de 0.5, y si se realiza una votación entre estos la eficacia será la misma ya que el resultado siempre coincide con el del clasificador D_3. Sin embargo, si se divide el espacio de representación en las tres regiones delimitadas por las líneas continuas, y para clasificar los objetos en la primera región (R_1) se utiliza el clasificador D_1, para clasificar los objetos en la segunda región (R_2) se utiliza el clasificador D_2, y para la tercera región (R_3) se utiliza el clasificador D_3, entonces la eficacia que se obtiene es la máxima.

Los métodos de selección de clasificadores se pueden dividir en dos tipos: los que seleccionan el clasificador más competente de manera dinámica durante la fase de clasificación y los que determinan las regiones de competencia de los clasificadores individuales durante una fase de entrenamiento. Los primeros a su vez se dividen en los que tienen en cuenta las salidas de los clasificadores y los que no. A continuación se expondrán varios ejemplos de todos estos métodos.

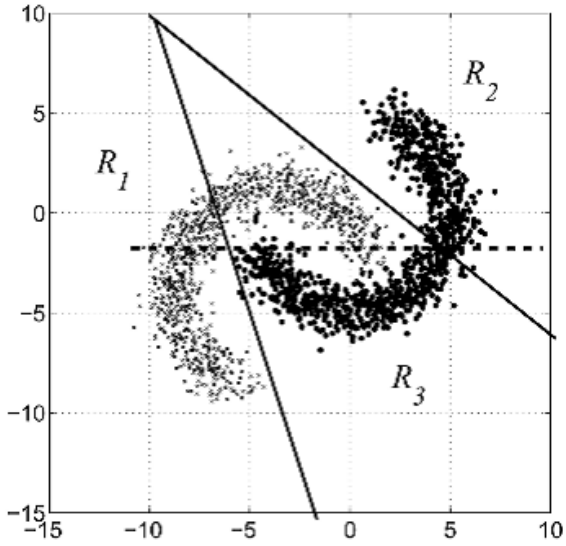

Fig. 2. Ejemplo de partición de un espacio de representación en el contexto de la selección de clasificadores.

3.4.1 Selección dinámica de los clasificadores

Los métodos presentados en esta sección, trabajan de la siguiente manera: una vez conocido el objeto representado por x que se debe clasificar, estiman cuál es el clasificador más competente para realizar esta tarea. Para lograr esto último se puede clasificar el objeto con los l clasificadores individuales y después, teniendo en cuenta las salidas de éstos, realizar la estimación, o solamente realizar la estimación a priori y solo clasificar el objeto con el clasificador de mayor estimación.

Estimación basada en los *k* vecinos más cercanos

Suponiendo que el espacio de representación sea un espacio métrico es posible estimar la competencia de los clasificadores individuales tomando los k vecinos más cercanos a x en el conjunto de entrenamiento o en algún conjunto de validación [52], midiendo la eficacia de los clasificadores en dicho conjunto, y entonces se manda a clasificar a x con el clasificador de más eficacia local estimada. Si se desea tener en cuenta las salidas de los clasificadores entonces se recomienda, para medir la competencia del clasificador D_i que dio como resultado la clase K_j, tomar los k vecinos más cercanos tales que para dichos elementos el resultado de D_i haya sido K_j, entonces la competencia de D_i es la razón entre la cantidad de elementos que verdaderamente pertenecían K_j y k.

Formalmente, sea V el conjunto de validación utilizado para las estimaciones, siendo $N_k(x)$ el conjunto que contiene los k vecinos más cercanos a x en V, la estimación de la competencia del clasificador D_i sin tener en cuenta su salida es:

$$comp(D_i, x) = \frac{\sum_{K_j \in \Omega} \sum_{y \in (N_k(x) \cap K_j)} I(D_i(x) = K_j)}{k}. \quad (16)$$

Donde I es una función indicadora. Por otra parte, siendo $N_k^{i,j}(x)$ el conjunto que contiene los k vecinos más cercanos a x en V tal que $y \in N_k^{i,j}(x) \rightarrow D_i(y) = K_j$, entonces si $D_i(x) = K_j$ su competencia estimada para clasificar al objeto representado por x teniendo en cuenta su salida es:

$$comp(D_i, x) = \frac{\left|N_k^{i,j}(x) \cap K_j\right|}{k}. \quad (17)$$

En caso de que el espacio de representación no fuera un espacio métrico es posible utilizar los k vecinos más similares [53], de acuerdo a alguna función de similitud definida. El parámetro k debe ser ajustado antes de empezar a utilizar el método, y es importante en este sentido tener en cuenta el tamaño del conjunto de validación V, pues si V es un conjunto pequeño y k es un número grande es posible que se utilicen elementos muy diferentes a x para medir la competencia de los clasificadores con x.

Este método presenta dos problemas fundamentales: solo utiliza información de tipo abstracta de la salida de los clasificadores y además no tiene en cuenta la distancia entre el objeto representado por x y sus k vecinos. En [54] es propuesto un método que da solución a estos problemas y se expone a continuación.

Estimación basada en la distancia a los k vecinos más cercanos

En el método propuesto en [54] se propone utilizar los soportes dados por los clasificadores individuales a la clase correcta aunque dicho soporte no haya sido el mayor. Para estimar la competencia del clasificador D_i se analizan los resultados obtenidos por éste al clasificar sus k vecinos más cercanos en V, en particular se promedian los soportes dados por D_i a las clases correctas en cada uno de los casos, en realidad se computa un promedio ponderado por las distancias entre cada uno de los k vecinos y el objeto representado por x. Utilizando las mismas notaciones que en la sección anterior la competencia de D_i en x quedaría formalmente definida como:

$$comp(D_i, x) = \frac{\sum_{K_j \in \Omega} \sum_{y \in (N_k(x) \cap K_j)} (D_i(x, K_j) * (1/d(x,y)))}{\sum_{y \in N_k(x)} 1/d(x,y)}. \quad (18)$$

En la ecuación anterior $D_i(x, K_j)$ denota el soporte dado por el clasificador D_i a la hipótesis $x \in K_j$, y $d(a, b)$ denota la distancia existente entre a y b.

Se procede análogamente para computar la competencia de un clasificador teniendo en cuenta su salida, solo que esta vez se tienen en cuenta los vecinos más cercanos que pertenecen a la clase que dio el clasificador como resultado. Formalmente, si $D_i(x) = K_j$, o sea el subconjunto K_j es el que recibe mayor soporte, entonces la competencia de D_i en x es:

$$comp(D_i, x) = \frac{\sum_{y \in N_k^{i,j}(x)} (D_i(x, K_j) * (1/d(x,y)))}{\sum_{y \in N_k^{i,j}(x)} (1/d(x,y))}. \quad (19)$$

Este método además de suponer espacio métrico, asume que todos los clasificadores dan un mismo tipo de soporte.

Ruptura de empates

Una vez estimadas las competencias de los clasificadores individuales con el objeto x, se da el resultado del clasificador con mayor competencia. ¿Pero qué sucede si existe un empate? Cuando existe un empate se pueden tomar varias opciones, entre las que se encuentran las siguientes:

- No seleccionar un solo clasificador, sino seleccionar varios y combinarlos de un modo distinto, por ejemplo utilizando un voto mayoritario.
- Intentar que el clasificador que tiene un segundo mayor grado de competencia realice el desempate, sería como realizar una votación entre los clasificadores que se encuentran empatados y el que les sigue en el orden de competencia.
- Tomar un clasificador de manera aleatoria.

Si se utiliza el primer enfoque hay que tener en cuenta el método de combinación a utilizar, pues por ejemplo si se utiliza una votación con un número par de clasificadores igual puede volver a ocurrir algún empate. Además, como no se conoce ni cuántos, ni cuáles clasificadores resultarán empatados puede ser muy difícil seleccionar un buen método de combinación a priori. La principal debilidad del segundo enfoque es que el segundo clasificador con mayor grado de competencia pudiera tener una eficacia muy mala. El tercer enfoque es arbitrario, y por lo tanto, la última opción a considerar.

3.4.2 Estimación previa de las regiones de competencia de los clasificadores

La selección dinámica de los clasificadores tiene el problema del costo computacional, por ejemplo, en los métodos anteriormente mostrados es necesario, cada vez que se va a clasificar un objeto, hallar los k objetos más cercanos, o similares a él en cierto conjunto. Otro enfoque consiste en dividir el espacio de representación en l subconjuntos propios $M_1 \subset S, \dots, M_l \subset S$ de forma tal que el clasificador D_i tenga una muy buena competencia con los objetos representados dentro del conjunto M_i, es decir, se trata de hallar un emparejamiento entre clasificador y región de competencia del mismo en la fase de entrenamiento. Nótese que crear una partición es conveniente para hacer las cosas más simples, si no se utiliza una partición, para un objeto se podrían seleccionar más de un clasificador y entonces habría que pensar también en cómo combinar estos. De esta forma, para clasificar un nuevo objeto solo habría que identificar dentro de qué subconjunto M_i se encuentra y utilizar el clasificador D_i para la tarea.

Agrupamiento

En [55] y [56] se propone correr algoritmos de agrupamientos sobre los objetos en el conjunto de entrenamiento y después asociar los clasificadores a los subconjuntos obtenidos. El funcionamiento del método en la fase de entrenamiento se describe en los siguientes pasos:

1- Se entrenan los l clasificadores individuales.
2- Sin tener en cuenta las clases a las que pertenecen, se agrupan a través de algún algoritmo de agrupamiento –por ejemplo *c-means*– los elementos del conjunto de entrenamiento en c subconjuntos, cada uno denotado por C_i.
3- Para cada subconjunto C_i se computan su elemento representativo $v_i \in S$ y el clasificador más eficaz con los objetos contenidos en él, denotemos dicho clasificador por $D_{(C_i)}$.

Entonces en la fase de clasificación el método solamente realiza los siguientes pasos:

1- Se halla el elemento más representativo v_i más cercano a x.
2- Se da como resultado $D_{(C_i)}(x)$.

Una variante de este método es hallar los subconjuntos de Z antes de entrenar los clasificadores, entonces el clasificador $D_{(C_i)}$ se entrena solamente con los objetos del subconjunto C_i haciendo de esta forma $D_{(C_i)}$ el clasificador experto en C_i.

Este método de selección depende mucho de cómo son obtenidos los subconjuntos C_i, y sobre todo de la coherencia que debe existir entre el algoritmo de agrupamiento utilizado y la estrategia para

obtener el elemento más representativo de cada subconjunto. Por ejemplo, si se utiliza el centro (*means*) como elemento más representativo entonces se estará suponiendo la existencia de una bola que contiene los elementos de C_i y se necesitará un algoritmo de agrupamiento como k-*means*.

4 Obtención de los clasificadores individuales

Una vez estudiados los principales métodos utilizados para combinar un conjunto de clasificadores, se estudiarán las estrategias más utilizadas para obtener los clasificadores individuales. Para ello se seguirá el siguiente procedimiento: primero se introducirá la estrategia general para entrenar los clasificadores y después se enumerarán los principales algoritmos en el estado del arte que utilizan dicha estrategia.

Como se ha venido observando, combinar clasificadores solo tiene sentido si estos no toman las mismas decisiones en todos los puntos. Otra forma de decir esto es que los errores cometidos por los clasificadores individuales no son los mismos, o sea que los clasificadores no se equivocan en los mismos puntos, de aquí que se complementen los unos a los otros. Lo antes planteado hace que la meta principal a la hora de obtener los clasificadores individuales que participarán en una combinación sea que estos se complementen entre sí.

Para crear un clasificador individual, se necesita un algoritmo de clasificación y un conjunto de entrenamiento, así, que si se desea crear clasificadores individuales que no cometan los mismos errores o al menos que sean distintos, necesariamente hay que variar el conjunto de entrenamiento o el algoritmo de clasificación. Estos dos enfoques serán estudiados en este capítulo.

Otro aspecto a tener en cuenta cuando se crean los clasificadores individuales, es si se realiza una división del problema general, en este caso, los clasificadores individuales van a resolver distintos problemas. Algunos autores consideran este aspecto como una modificación del conjunto de entrenamiento, lo que en lugar de modificar los rasgos se modifica las clases a la que pertenece cada objeto. Este enfoque también será desarrollado en este capítulo.

4.1 Manipulación del conjunto de entrenamiento

Sin dudas la estrategia más popular para generar clasificadores individuales es la variación en las muestras de entrenamiento que serán utilizadas en el entrenamiento de cada uno de ellos. Dicha variación puede ser llevada a cabo de distintas maneras, por ejemplo, se pueden utilizar tanto objetos distintos como rasgos distintos. Además se pueden variar los elementos eliminándolos o incluyéndolos del conjunto de entrenamiento del clasificador individual en cuestión o simplemente asignándole un peso que codifique su importancia en ese conjunto.

4.1.1 Bagging

Si se quisiera obtener clasificadores individuales que no cometan los mismos errores, una opción interesante pudiera ser entrenar cada uno de los clasificadores con una muestra diferente. Si se tiene la posibilidad de obtener un conjunto de entrenamiento de gran tamaño, lograr esto no es un problema ya que se podría dividir el conjunto varias veces, Sin embargo, muchas veces en la práctica se cuenta con un conjunto de entrenamiento limitado. En este caso, una variante es construir l conjuntos de entrenamientos a partir de la realización de muestreos con reemplazo del conjunto original. Nótese que de esta forma cada conjunto puede contener elementos repetidos.

Leo Breiman propone en [57] el algoritmo Bagging como una estrategia de obtención de los clasificadores individuales de un esquema. El término Bagging proviene de la frase en inglés ***B**ootstrap **Agg**regat**ing***, que deja claro que el método combina los resultados de clasificadores entrenados con conjuntos de entrenamientos generados por la técnica de *bootstraping* [58], es decir se crean conjuntos de entrenamientos realizando muestreos con reemplazos de un conjunto original. Cada clasificador

individual es obtenido a partir de un algoritmo de clasificación que se denotará por A y que es parámetro del método Bagging, luego en la etapa de clasificación se combinan los resultados a través de una votación y siguiendo el consenso de la pluralidad.

A continuación se describirá el funcionamiento de Bagging en la etapa de entrenamiento de una manera más formal:

1- Para el entrenamiento, Bagging recibe como parámetros el conjunto de entrenamiento Z, un algoritmo de clasificación A, y un número natural l que representa el número de iteraciones, esto es el número de clasificadores individuales a crear.
2- Se crean exactamente l conjuntos de entrenamiento, denotados por $B_1, B_2, \dots, B_l$, todos del mismo tamaño que Z, generados a partir de un muestreo con reemplazo [58] de este último.
3- Por cada B_i se crea un clasificador individual utilizando el algoritmo de clasificación A, o sea $D_i = A(B_i)$.

En la fase de clasificación todos los clasificadores individuales dan su voto –en caso de que las salidas no sean abstractas se toma el mayor soporte o el mejor rango–, la clase con mayor cantidad de votos es el resultado final y los empates se resuelven aleatoriamente.

Para que los clasificadores B_i no estén muy correlacionados entre sí –no den casi siempre los mismos resultados– es necesario que el algoritmo de clasificación A sea sensible a ligeros cambios en el conjunto de entrenamiento. Breiman introduce entonces, en el trabajo antes citado, el término de la estabilidad del algoritmo de clasificación de una manera heurística.

Definición 3 (Estabilidad de un algoritmo de clasificación según Breiman). Un algoritmo de clasificación es inestable si a ligeros cambios en el conjunto de entrenamiento se obtienen cambios significativos en la clasificación.

En realidad la definición dada por Breiman es muy relativa ya que el significado de cambios significativos puede variar según el contexto. Buhlmann propone una definición más formal y general de la propiedad de estabilidad en [59] aunque esta –según sus autores–, no es inconsistente con la definición propuesta por Breiman.

Entre los algoritmos de clasificación que Breiman clasifica como inestables están los que utilizan árboles de decisión, redes neuronales y entre los estables los algoritmos de vecinos más cercanos y los que usan funciones de discriminante lineal [60].

Otra de las ventajas que aporta el utilizar muestreos con reemplazo es la siguiente: según [61] en cada B_i se quedan fuera aproximadamente un 37% de los objetos. Esto se debe a que la probabilidad de que un elemento del conjunto de entrenamiento sea seleccionado cierta cantidad de veces distribuye aproximadamente Poisson con $\lambda = 1$. En este sentido dichos objetos se pueden aprovechar para realizar una mejor estimación del error de generalización.

Sea O_i el conjunto de objetos de Z que no están presentes en la muestra B_i, nótese que el clasificador D_i no ha sido entrenado con ninguno de los elementos presentes en O_i. Sea $x \in Z$, y sea un esquema D obtenido a través de la estrategia Bagging, con clasificadores individuales $D_i = A(B_i)$, se define la clasificación *out-of-bag* del objeto representado por x a través de D –y se denota por $D^{oob}(x)$ – como el resultado de la clasificación de x con el esquema D pero solo teniendo en cuenta el resultado de los clasificadores individuales D_i tales que $x \in O_i$. Más formalmente:

$$D^{oob}(x) = K_i,\ i = \max_{1 \le j \le k}\left(\sum_{1 \le i \le l}\left(I\left(D_i(x) = K_j\right) * I(x \in O_i)\right)\right). \quad (20)$$

En la ecuación anterior se utilizan dos funciones indicadoras, la primera $I\left(D_i(x) = K_j\right)$ da como resultado 1 si el clasificador D_i asigna el objeto a la clase que es tenida en cuenta en ese momento y 0 en otro caso, la segunda $I(x \in O_i)$ da como resultado 1 si el objeto x no pertenece al conjunto con que fue entrenado D_i y 0 en otro caso. De esta forma, a partir de D^{oob} es posible estimar de una forma más certera el error de generalización de D.

Según Breiman D^{oob} puede ser utilizado en otros contextos, por ejemplo para estimar las probabilidades a posteriori de cada una de las clases una vez conocido el objeto representado por x, esto es $P(K_i|x)$. Cuando se utilizan árboles de decisión como clasificadores individuales se puede estimar las probabilidades a posteriori de las clases en cada una de las hojas del árbol. Breiman en [61] muestra de manera empírica cómo la estimación de estas probabilidades a través de D^{oob} se acercan más a su valor real.

Bagging ha demostrado tener un gran éxito cuando el algoritmo de clasificación utilizado es inestable, por ejemplo Breiman reporta disminución en los promedios de clasificación incorrecta de hasta un 43% en problemas con una cantidad pequeña de muestras y hasta de un 77% con una cantidad de muestras grande [57]. A continuación se expondrá la argumentación de Breiman con respecto al éxito de Bagging.

Sea $B = \{B_1, B_2, \dots, B_l\}$ el conjunto de muestras de Z computadas por el algoritmo Bagging. Se define entonces la función $Q: (A, x, B, K_j) \rightarrow [0,1]$, que recibe como parámetros un algoritmo de clasificación A, un objeto representado por x, un conjunto de conjuntos de entrenamientos B, y una clase K_j; y da como resultado la probabilidad que tiene el algoritmo de clasificación A de clasificar –o de generar un clasificador que clasifique– al objeto representado por x como perteneciente a la clase K_j a través de todos los conjuntos de entrenamiento en B. O sea:

$$Q(A, x, B, K_j) = \frac{\sum_{i=1}^{l} I(A(B_i, x) = K_j)}{|B|}. \quad (21)$$

Luego la probabilidad que tiene el algoritmo de clasificación A de clasificar, a través de los conjuntos de entrenamiento en B, correctamente al objeto representado por x, es la suma de las probabilidades de clasificación correcta con cada una de las clases, o sea:

$$\sum_{j=1}^{k} (Q(A, x, B, K_j) * P(K_j|x)). \quad (22)$$

Bagging también puede ser estudiado como un algoritmo de clasificación, entonces quedaría definido como $A^B(Z, x) = \text{argmax}_{K_j \in \Omega} Q(A, x, B, K_j)$, nótese que esta definición es equivalente a la dada anteriormente pues siempre se da como resultado la clase que más votos recibe a través de los clasificadores generados a partir de las muestras B_i. Por la forma en que está definido A^B, la probabilidad de que clasifique correctamente un objeto representado por x queda definida por:

$$\sum_{j=1}^{k} (I\left(\text{argmax}_{K_j \in \Omega} Q(A, x, B, K_j) = K_j\right) * P(K_j|x)). \quad (23)$$

Teniendo en cuenta todos los objetos representados en S, la probabilidad de clasificación correcta de A^B se define como:

$$\int \left[\sum_{j=1}^{k} (I\left(\text{argmax}_{K_j \in \Omega} Q(A, x, B, K_j) = K_j\right) * P(K_j|x))\right] P(x). \quad (24)$$

En caso de que S sea finito se sustituyen las integrales por sumatorias en la ecuación anterior. Breiman dice que un algoritmo de clasificación tiene un orden correcto –utiliza el término *order-correct* en inglés– si a través de los conjuntos de entrenamiento en B el algoritmo asigna x a la clase correcta la mayoría de las veces. Formalmente:

Definición 4. Un algoritmo de clasificación A tiene un *orden correcto* con un conjunto de conjuntos de entrenamiento B en un objeto representado por x si $\text{argmax}_{K_j \in \Omega} Q(A, x, B, K_j) = \text{argmax}_{K_j \in \Omega} P(K_j|x)$.

Es importante notar que un algoritmo A puede tener un orden correcto con B en x y sin embargo no ser un algoritmo eficaz. Breiman pone el siguiente ejemplo. Sea un problema con dos clases K_1 y K_2 tal

que $P(K_1|x) = 0.9$ y $P(K_2|x) = 0.1$. Puede ocurrir que $Q(A, x, B, K_1) = 0.6$ y $Q(A, x, B, K_1) = 0.4$ por lo que se tiene que A tiene un orden correcto con B en x y sin embargo, una probabilidad de clasificación correcta de 0.58, cuando el clasificador Bayesiano –óptimo– tiene una probabilidad de clasificación correcta de 0.9.

Nótese que si el algoritmo de clasificación A tiene un orden correcto con B en x entonces la probabilidad de clasificación correcta de A^B en ese punto sería $\max_i P(K_i|x)$, es decir, A^B sería un clasificador óptimo –en el sentido Bayesiano– en el punto x.

Es posible dividir el espacio de representación S en dos subconjuntos disjuntos: el conjunto C que contendrá todos los x en los cuales el algoritmo A tiene un orden correcto con B y C^c el complemento de C. De esta forma la probabilidad de clasificación correcta de A^B en todo S puede escribirse:

$$\int \max_i P(K_i|x)\, P(x) + \int \left[\sum_{j=1}^{k} (I\left(\text{argmax}_{K_j \in \Omega}\, Q(A, x, B, K_j) = K_j\right) * P(K_j|x))\right] P(x). \quad (25)$$

Donde el la primera integral es sobre los puntos que pertenecen a C y la segunda sobre los putos que no pertenecen a C.

Es decir, Bagging crea un clasificador óptimo en los puntos en los que el algoritmo de clasificación original A tiene un orden correcto, incluso aunque este último no sea eficaz. Luego un algoritmo de clasificación que tenga un orden correcto en la mayoría de los puntos del espacio de representación puede generar clasificadores que se aproximan al óptimo –como es order-correct en casi todos los puntos y no en todos hay puntos en los que no es order-correct, entonces en estos puntos el clasificador no es óptimo por eso se dice que el algoritmo se aproxima al óptimo–, utilizando Bagging.

En otras palabras, lo que nos está diciendo Breiman es que si se tiene un clasificador no sesgado no parcializado pero que tiene suficiente varianza, aplicarle Bagging debe dar como resultado clasificadores parecidos al óptimo, pues se reduce la varianza sin añadir sesgo. Por otra parte se debe notar que lo que plantea Breiman sirve para hacerse una idea intuitiva acerca del funcionamiento de Bagging pero carece de utilidad práctica alguna, pues es imposible saber cuándo un algoritmo de clasificación tiene un orden correcto en un punto o región dada [62].

Se debe notar que si el algoritmo de clasificación A no tiene un orden correcto en un objeto no se garantiza que Bagging mejore la precisión de A en ese punto. De hecho, según afirma Breiman aplicarle Bagging a un algoritmo que no tiene un orden correcto en la mayoría de los puntos del espacio de representación es una mala idea. También es una mala idea aplicarle Bagging a un algoritmo de clasificación estable.

Otros enfoques para el estudio de Bagging han sido propuestos, por ejemplo [59] y [63] utilizan enfoques estadísticos, mientras que en [62] se analiza desde una perspectiva Bayesiana. Sin embargo, y como se reconoce en [64], aún sigue faltando un entendimiento claro y útil de la eficacia de Bagging.

Recientemente en [64] es propuesta otra metodología, esta consiste en comparar el algoritmo Bagging a lo que se define como Bagging ideal. En el Bagging ideal los conjuntos de entrenamiento B_i son tomados directamente de la distribución del problema. En dicho trabajo se propone la utilización de otro algoritmo para la generación de los B_i que exponemos a continuación:

1- El algoritmo recibe como parámetros el conjunto de entrenamiento Z, el número de conjuntos a generar l y el tamaño deseado para esos conjuntos b.
2- Se crean l conjuntos vacíos $B_1, B_2, \dots, B_l$.
3- Si $\forall i(1 \leq i \leq l)[|B_i| = b]$ se termina el algoritmo.
4- Se toma un B_j de manera aleatoria de entre los B_i con menor cantidad de elementos.
5- Se toma $x \in Z$ tal que $x \notin B_j$ de forma aleatoria de entre los elementos de Z que aparecen con menor frecuencia en los conjuntos B_i.
6- Se agrega x a B_j, o sea $B_j = B_j \cup \{x\}$.
7- Se regresa al paso 3.

Cuando se combinan la forma de generación de los subconjuntos B_i presentada en el algoritmo anterior con la combinación de los resultados de los clasificadores $D_i = A(B_i)$ a través de la votación con la pluralidad como forma de consenso, se obtiene el algoritmo –propuesto en [64]– Bagging diseñado (*Bagging-by-design*). El núcleo central del trabajo gira en torno al impacto que tienen las intersecciones en los conjuntos B_i, de hecho el algoritmo anteriormente mostrado intenta minimizar estas intersecciones. Bajo la hipótesis de que a una mayor intersección entre los conjuntos B_i implica una mayor correlación entre los clasificadores individuales D_i se muestra evidencia, tanto teórica como empírica, que la diferencia desde el punto de vista estadístico entre el Bagging ideal y el Bagging diseñado es menor que la diferencia entre el Bagging ideal y el Bagging original. Este es un trabajo reciente y es difícil todavía medir su impacto.

4.1.2 Método del subespacio aleatorio

En [65] es propuesto un algoritmo para la creación de distintos árboles de decisión, con estos árboles se crea un esquema que da como resultado final los promedios de las probabilidades a posteriori de las clases en las hojas en las que caen los objetos a clasificar en cada uno de los árboles; dicho esquema recibe el nombre de bosque de decisión.

Sea n el número de dimensiones del espacio de representación de los objetos a clasificar. Sea el conjunto $R = \{R_1, R_2, \dots, R_n\}$ que contiene los dominios de definición de los rasgos que son observados en cada objeto, o sea, como ya se ha visto el espacio de representación no es más que el producto cartesiano de dichos conjuntos. Se define entonces la función f con dominio en $(S \times 2^R)$, y que al tomar un objeto $[x = (r_1, \dots, r_n)] \in S$ y un $R' \in 2^R$ (2^R es el conjunto potencia de R) devuelve como resultado una tupla $x' = (r_1', \dots, r_{|R'|}')$ la cual es creada a partir de x eliminando los rasgos que no están presentes en R'. O sea f selecciona los rasgos presentes en R' de x. El algoritmo propuesto por Ho es el siguiente:

1- El algoritmo recibe como parámetros un conjunto de entrenamiento Z, un algoritmo de clasificación A que utilice árboles de decisión, y un número l que representa el número de iteraciones a realizar, esto es el número de clasificadores a generar.
2- Se seleccionan aleatoriamente sin reemplazo l conjuntos $R_1', R_2', \dots, R_l'$ de 2^R.
3- Se generan l conjuntos $B_1, B_2, \dots, B_l$ tales que $B_i = f(Z, R_i')$. O sea cada B_i es la imagen de f con todos los elementos de Z y R_i'.
4- Se generan l clasificadores –árboles de decisión– $D_1, \dots, D_l$ tales que $D_i = A(B_i)$.

Como se dijo anteriormente, los resultados obtenidos a partir de los árboles de decisión son combinados utilizando la función promedio asumiendo que los árboles dan salidas de tipo medida.

Ho realiza varias comparaciones para evaluar el método propuesto. Cuando compara con árboles de decisión simples, el método resulta ser superior excepto en los casos en que el espacio de representación tiene una dimensión pequeña, por ejemplo cuando $n = 9$. Se compara también contra esquemas de árboles de decisión generados a partir de Bagging y Boosting para analizar las diferencias entre los árboles que conforman los esquemas, en este sentido anota que el método propuesto genera árboles más diferentes entre sí que los generados por Bagging y Boosting.

La principal ventaja del método es que donde otros algoritmos ven un problema –la maldición de las grandes dimensiones– este ve una oportunidad, así que cuando se esté en presencia de problemas con grandes dimensiones se tiene una opción. Otra buena propiedad que presenta el método es que es independiente al algoritmo de clasificación, pues aunque estos –según Ho– tengan que ser sobre árboles de decisión no tiene que ser alguno en específico.

Pero, resulta que la mayor ventaja será también la mayor desventaja pues con problemas de pocas dimensiones el método no funciona bien, según observa Ho. Otro aspecto negativo a tener en cuenta es la selección de los subconjuntos de rasgos, pues se realiza con poco control ya que no se tiene en cuenta la calidad de la selección de estos, o sea, cabe la posibilidad que un clasificador del esquema se entrene con un conjunto de rasgos que no represente el problema de forma adecuada.

4.1.3 Attribute Bagging.

En realidad el método de subespacio aleatorio no tiene por qué usarse solamente con árboles de decisión, y en este sentido en [66] fue propuesto esencialmente el mismo algoritmo pero con algunas diferencias, una de ellas es el nombre: (*Attribute Bagging*). En este método se estima de antemano un número $b < n$, –recuérdese que n es la cantidad de dimensiones del espacio de representación– que representa la cantidad de rasgos a tener en cuenta en cada conjunto de entrenamiento B_i. El valor de b se elige en dependencia de características propias del domino del problema en cuestión o experimentalmente analizando la eficacia de clasificadores generados con distintos valores de b. Al principio se pueden generar más B_i que los que van a ser utilizados ya que los l conjuntos de entrenamiento finales serán tomados a partir de la eficacia del algoritmo A en cada uno de ellos. Los resultados se combinan por votación.

En [66] se realizó una comparación entre el Attribute Bagging y el Bagging original ambos utilizando árboles de decisión donde se observa una superioridad del primero sobre el segundo, y esto ocurre sobre todo cuando la cantidad de clasificadores individuales comienza a aumentar. Es importante anotar que para dicha comparación solamente se utilizó una base de datos con $d = 28$. Nótese que cuando existe mucha redundancia entre los rasgos existe la posibilidad de que los clasificadores entrenados con un subconjunto de los rasgos tengan una eficacia mayor que el clasificador que se entrena con todos; si a esto se le suma de que los clasificadores individuales en el método de Attribute Bagging se entrenan con un número mayor de objetos que en el Bagging original no extrañaría el resultado anterior.

Según aumenten las dimensiones del problema y disminuya la cantidad de objetos disponibles para entrenar, el método Attribute Bagging debe ser preferido sobre el Bagging original. Si se compara el Attribute Bagging con el método de subespacio aleatorio el primero tiene a favor que tienen en cuenta la calidad de los subconjuntos de rasgos a utilizar.

4.1.4 Random Forest

En el año 2001, Leo Breiman, influenciado por los métodos de subespacio aleatorio [65] y por el trabajo [67] de Dietterich, propone el algoritmo *Random forest* [68]. El algoritmo consiste en crear un esquema de árboles de decisión introduciendo aleatoriedad en la selección de muestras de entrenamiento y en el mismo proceso de formación del árbol.

Para un mayor entendimiento del algoritmo se dará una breve introducción a cómo funcionan los algoritmos de clasificación con árboles de decisión. A grandes rasgos estos algoritmos funcionan creando particiones recursivas sobre el espacio de representación teniendo en cuenta los elementos en el conjunto de entrenamiento, cada nodo representa una parte de la partición superior, sus hijos dividirán a su vez la parte que él representa. Si los elementos en un nodo son lo suficientemente representativos de una clase, –esto se puede saber a partir de un grado de pureza por ejemplo–, entonces ese nodo deberá ser una hoja, y representará la clase en cuestión. En la etapa de clasificación el elemento a clasificar recorre el árbol desde la raíz hasta alguna hoja dando como resultado la clase a la cual representa dicha hoja.

Los algoritmos de clasificación con árboles de decisión varían la forma en que se verifica si una clase es representada o no por un nodo, la forma en que realizan las particiones, entre otras. La mayoría de las veces estos algoritmos realizan la partición a partir de un rasgo, es decir ellos evalúan todos los rasgos de los objetos y determinan cuál es el mejor para dividir el conjunto, por ejemplo $r_i < c$ es una condición para dividir un conjunto de elementos en dos.

A continuación se explica el algoritmo propuesto por Breiman:

1- El algoritmo recibe como parámetros un conjunto de entrenamiento Z, un algoritmo de clasificación A, un entero l que codifica el número de iteraciones, y un número natural $m \leq n$.
2- Se generan l conjuntos $B_1, \ldots, B_l$ realizando $|Z|$ muestreos con reemplazo de Z (el mismo procedimiento que en Bagging).

3- Se crean l árboles de decisión $D_1, \dots, D_l$ haciendo $D_i = A(B_i)$, pero, teniendo en cuenta que para realizar la partición en cada uno de los nodos se tienen en cuenta solamente un conjunto aleatorio de rasgos R', donde $|R'| = m$.

La combinación final de los resultados de los árboles individuales se realiza mediante votación con la pluralidad como consenso. Otro aspecto a tener en cuenta es que en el algoritmo no se realizan podas, es decir, cada árbol se deja crecer lo máximo posible.

Se puede notar que Bagging con árboles de decisión es un caso particular de *Random forest*, cuando $m = d$. Aunque la semejanza con Bagging hace pensar que el algoritmo debe reducir la varianza, en [68] se menciona que la eficacia obtenida por el algoritmo indica que este debe reducir la parcialidad. En una gran variedad de problemas se ha mostrado que *Random forest* es superior a los respectivos árboles de decisión.

El algoritmo *Random forest* ha recibido una gran atención en la comunidad científica por su eficacia, varias variantes del algoritmo han sido propuestas, dos de las más recientes son [69] y [70]. En general las variantes introducen modificaciones en la manera en que se introduce la aleatoriedad o simplemente ajustes con respecto al domino del problema en cuestión. Debido a esta gran popularidad del método *Random forest* ya es considerado en la literatura como un método más general que el propuesto originalmente por Breiman, Random forest se considera entonces un esquema de árboles de decisión donde la aleatoriedad se introduce de algún modo.

Pese al auge del método, su estudio desde el punto de vista teórico ha recibido muy poca atención, y esto se debe a la alta dificultad que conlleva plantear un análisis formal del mismo [71]. En este sentido la comunidad ha seguido el enfoque de presentar versiones simplificadas del algoritmo y entonces analizarlas. Con el paso de los años dichas versiones se han ido acercando al algoritmo original, sin embargo estas aún no alcanzan ni el original ni las versiones de éste utilizadas en la práctica. Son de destacar aquí los trabajos siguientes, que siguen una línea común de desarrollo: [72], [73], [74], [75], [71].

Es importante señalar que los estudios teóricos sobre el método se han concentrado sobre todo en el problema de la consistencia, esto es, que el esquema se haga un clasificador óptimo cuando el tamaño de la muestra de entrenamiento tienda a infinito. Pero si bien el problema de la consistencia es importante, consideramos deberían tener prioridad estudios que den soporte a razonamientos matemáticos que expliquen otras cuestiones, por ejemplo: ¿por qué es superior Random forest a las versiones simples de los árboles de decisión?, ¿qué factores son los que influyen, y en qué medida lo hacen, en dicha superioridad? Esas y otras preguntas, más de una década después de creado el método siguen sin ser respondidas de una manera rigurosa.

4.1.5 Boosting

Boosting es una metodología para la creación de esquemas de clasificación que intenta construir un clasificador eficaz combinando los resultados de un grupo de clasificadores *débiles*. Intuitivamente, por clasificador débil se entenderá un clasificador que tiene una probabilidad de clasificación correcta solamente un poco superior a la de un clasificador aleatorio. En otras palabras, se tiene un algoritmo de clasificación A del que se asume solamente que puede producir clasificadores débiles –nótese que dichos clasificadores no tienen por qué ser eficaces, por ejemplo, para el caso en que se tienen 2 clases, pudieran tener solamente una probabilidad de clasificación correcta de 0.51– el objetivo es a partir de A crear un clasificador eficaz.

Para lograr su objetivo los algoritmos que siguen la metodología de Boosting crean clasificadores a partir de A secuencialmente de forma tal que cada clasificador se enfoque más en los objetos en los cuales los clasificadores anteriormente creados han tenido más problemas. Pero, ¿cómo hacer que los clasificadores se concentren más en unos objetos que en otros? Si se codifica la importancia de cada objeto en Z a través de una distribución de pesos $Z^{\mathcal{D}}$ existen dos enfoques para lograrlo:

- En caso de que el algoritmo de clasificación A sea capaz de manejar esta cuestión internamente se le pasará como parámetro además del conjunto de entrenamiento Z, la distribución de pesos

$Z^{\mathcal{D}}$. Entonces $A(Z, Z^{\mathcal{D}})$ denota el clasificador producido por el algoritmo A con el conjunto de entrenamiento Z dándole una importancia a cada objeto en Z según $Z^{\mathcal{D}}$.

- En caso de que el algoritmo de clasificación no tenga la capacidad de tener en cuenta a $Z^{\mathcal{D}}$ se realizará un re-muestreo de Z según $Z^{\mathcal{D}}$.

A grandes rasgos, la metodología Boosting puede ser definida como sigue, nótese que lo que sigue no es un algoritmo completamente definido, sino una metodología general:

1- El algoritmo recibe como parámetros un algoritmo de clasificación A, un conjunto de entrenamiento Z, una distribución de pesos inicial $Z^{\mathcal{D}}$, un número de iteraciones l.
2- Se toma como distribución inicial a $Z^{\mathcal{D}}$, es decir, se hace $Z_1^{\mathcal{D}} = Z^{\mathcal{D}}$.
3- Se realizan l iteraciones (con i desde 1 hasta l).
4- Se entrena el clasificador i-ésimo, o sea $D_i = A(Z, Z_i^{\mathcal{D}})$.
5- Si $i = l$ se termina el entrenamiento, y el resultado es un esquema con los clasificadores individuales D_i que realiza la combinación de sus resultados de alguna manera.
6- Se evalúa el desempeño de D_i en Z.
7- Se crea la distribución de pesos $Z_{i+1}^{\mathcal{D}}$ de modo que se ajuste al desempeño de los clasificadores anteriormente entrenados $D_1, \dots, D_i$.
8- Se incrementa i y se regresa al paso 4.

La metodología de Boosting definida anteriormente no es todavía un algoritmo, pues no está definido completamente, por ejemplo aún no se especifica cómo combinar los clasificadores, o cómo ir actualizando las distribuciones en cada iteración. El primer algoritmo Boosting fue primeramente planteado por Schapire [76] en respuesta a una pregunta de teoría de aprendizaje computacional planteada en [77]. Dicha pregunta, explicada a grandes rasgos, se interesaba por saber si era posible volver eficaz a un clasificador débil. La respuesta dada por Schapire era positiva, y correcta desde el punto de vista teórico, pero tenía ciertas deficiencias que impedían su utilización en la práctica. Ya en [78] se propone el algoritmo Adaboost que es el arquetipo de los algoritmos de tipo Boosting, ya que fue el primero en ser propuesto, es eficiente computacionalmente, y ha demostrado su eficacia en un sinnúmero de aplicaciones en la práctica.

Adaboost es un algoritmo desarrollado para resolver problemas de clasificación con solamente dos clases. A continuación se expondrá el algoritmo:

1- El algoritmo recibe como parámetros un conjunto de entrenamiento Z, un algoritmo de clasificación A y un número de iteraciones a realizar l.
2- Primeramente se inicializa la distribución de pesos para dar la misma importancia a todos los elementos de Z. Esto es $\forall x \in Z[Z_1^{\mathcal{D}}(x) = {}^{1}/_{|Z|}]$.
3- Se realizan l iteraciones (con i desde 1 hasta l)…
4- Se entrena el clasificador i-ésimo, o sea $D_i = A(Z, Z_i^{\mathcal{D}})$.
5- Se calcula la probabilidad de error de D_i en Z, esto es $e_i = \frac{\sum_{x \in Z} I(\neg(D_i(x) = K_j \wedge x \in K_j))}{|Z|}$ donde I es una función indicadora. Nótese que si $e_i \geq 0.5$ entonces D_i no es un clasificador débil.
6- Se computa $\alpha_i = \frac{1}{2}\ln(\frac{1-e_i}{e_i})$ que será el peso asociado al clasificador D_i en la función de combinación resultante (como se verá en el paso siguiente).
7- Si $i = l$ se termina el entrenamiento, y el resultado es un esquema con los clasificadores individuales D_i que combina los resultados de estos mediante una votación mayoritaria ponderada según los pesos α_i. Sino el algoritmo continúa.

8- Se crea la distribución de pesos $Z_{i+1}^{\mathcal{D}}(x) = \frac{Z_i^{\mathcal{D}}(x) * \lambda(x)}{\sigma}$ donde $\sigma = 2[e_i(1-e_i)]^{1/2}$ es un factor de normalización y $\lambda(x) = \begin{cases} e^{\alpha}, si(D_i(x) = K_j) \wedge (x \in K_j) \\ e^{-\alpha}, en\ cualquier\ otro\ caso \end{cases}$.

9- Se incrementa i y se regresa al paso 4.

Una de las formas de explicar el algoritmo Adaboost es partiendo del algoritmo general de Boosting utilizando como función de combinación la votación mayoritaria ponderada, y entonces obtener tanto los pesos de los clasificadores en la combinación (es decir los α_i) como la estrategia para actualizar las distribuciones de los pesos ($Z_i^{\mathcal{D}}$) de forma tal que se minimice la función de pérdida exponencial (dicha función será explicada a continuación).

Como se está trabajando con problemas de clasificación simple (cada objeto pertenece a solo una clase) y asumiendo que del resultado de los clasificadores se tendrá en cuenta su salida abstracta, se puede considerar una función de pérdida $\xi: (\Omega \times \Omega) \to \mathbb{R}$, es decir, $\xi(K_i, K_j)$ representa el costo asociado al hecho de que un clasificador dé como salida la clase K_i cuando el objeto pertenece a K_j. La función de pérdida exponencial se define de la siguiente manera:

$$\xi_{exp}(K_i, K_j) = \begin{cases} \exp(1), \ si\ K_i = K_j \\ \exp(-1), en\ otro\ caso \end{cases}. \quad (26)$$

En [79] y [32] se muestra cómo la forma en que se asignan los pesos de los clasificadores en la combinación y la estrategia seguida para ir actualizando la distribución de los pesos de los objetos en el conjunto de entrenamiento, minimizan la función de pérdida exponencial en cada iteración del algoritmo. Aunque el algoritmo Adaboost fue originalmente ideado para el caso en que se tienen solamente 2 clases, este fue extendido casi de inmediato para soportar cualquier cantidad de clases, por ejemplo los algoritmos Adaboost.M1 y Adaboost.M2.

En [78] se demuestra el siguiente teorema para cuando el problema tiene solamente 2 clases (y en [1] se propone un teorema análogo para el caso con más de dos clases):

Teorema 3. Sean $\Omega = \{K_1, K_2\}$, e el error del esquema producido por Adaboost en el conjunto de entrenamiento Z (el error de entrenamiento), e_i el error del clasificador individual D_i en Z, ponderado por la distribución de pesos con que fue entrenado D_i, o sea $Z_i^{\mathcal{D}}$. Se cumple entonces que:

$$e < 2^l \prod_{i=1}^{l} \sqrt{e_i(1-e_i)}. \quad (27)$$

El Teorema 3 da una cota superior al error del esquema con los objetos de Z y demuestra que cuando el número de clasificadores aumenta el error se aproxima a cero. Pero ¿qué sucede con el error del esquema con elementos no presentes en Z, es decir, con el error de generalización? La intuición dice que según aumente l dicho error debe disminuir hasta un punto y después volver a crecer debido al sobreajuste [80]. La intuición se corresponde con muchos casos de la realidad ya que los algoritmos de Boosting pueden llegar a sobreentrenar [81], pero en muchos casos el error con objetos de prueba sigue disminuyendo aun cuando el error con elementos de Z ha alcanzado el valor cero y el número de iteraciones supera las mil [80].

Según afirma Rob Shapire entre las ventajas del algoritmo Adaboost y de la mayoría de los algoritmos del tipo Boosting se pueden contar las siguientes, las cuales son naturalmente aceptadas por toda la comunidad:

- Es computacionalmente eficiente, por ejemplo, el algoritmo de detección de rostros propuesto por Viola y Jones [82] utiliza Boosting y es uno de los algoritmos más famosos y eficientes en su campo.
- Es simple y fácil de implementar.
- No tiene parámetros que ajustar más allá del número de iteraciones.

- Puede ser utilizado con distintos algoritmos de clasificación.
- Cuenta con una sólida teoría detrás que le da soporte.

Y entre las desventajas muchos autores coinciden en que se pueden contar:

- Cuando el algoritmo de clasificación es muy complejo (o sea se ajusta demasiado a los datos) entonces el algoritmo tiene cierta tendencia al sobre-entrenamiento.
- El algoritmo es muy sensible al ruido y tiende a fallar en presencia de éste.

Ha sido observado experimentalmente en [83] que cuando se le aplica Boosting a un algoritmo de clasificación, en las primeras iteraciones se reduce el sesgo y después se reduce la varianza.

4.2 Variaciones en los algoritmos de clasificación

Otra estrategia muy popular para crear sistemas de múltiples clasificadores es la de variar el (los) algoritmo(s) de clasificación. Para crear l clasificadores individuales se puede utilizar solamente un algoritmo de clasificación o varios, por lo que existen dos formas básicas de generar clasificadores distintos a través de variaciones en los algoritmos de clasificación. La primera es utilizando un solo algoritmo de clasificación pero que este contenga cierto factor interno que varíe –por ejemplo *Random forest* varía aleatoriamente los rasgos– que cause diferencia en los clasificadores que se vayan creando y la segunda es utilizando distintos algoritmos de clasificación. A continuación se expondrán dichas variantes.

4.2.1 Variación en el algoritmo de clasificación

Existen variantes del algoritmo *Random forest* que no realizan un muestreo del conjunto de entrenamiento [84] es decir, los árboles se entrenan todos con los mismos objetos en Z. En este caso la diferencia entre los árboles está siendo causada por un factor de aleatoriedad presente en el algoritmo –la selección aleatoria de rasgos para expandir un nodo– por lo que estas variantes de Random forest bien pudieran ser consideradas métodos que trabajan a través de variaciones en el algoritmo de clasificación.

Otro enfoque muy utilizado en este sentido es el de combinar varias redes neuronales. Como bien se explica en [85] cuando se entrena una red neuronal se pueden obtener óptimos locales, dado que los pesos iniciales de los nodos se asignan de manera aleatoria, a través de distintas redes neuronales se pueden tener distintos óptimos locales, por los que dichas redes no tienen por qué cometer los mismos errores. Luego si los resultados de varias redes neuronales se combinan adecuadamente se puede obtener un desempeño superior al obtenido por la utilización de una sola red neuronal.

Como todos los clasificadores individuales son generados a partir de un mismo algoritmo de clasificación y con un mismo conjunto de entrenamiento, modelar el esquema debería ser una tarea fácil, facilitando así el estudio de las propiedades de este. Sin embargo, el factor que varía dentro de estos en algunos casos es aleatorio –por ejemplo en *Random forest*– y actúa como contraparte complicando demasiado el modelo y haciendo difícil su entendimiento.

4.2.2 Utilización de distintos algoritmos de clasificación

Cada algoritmo de clasificación tiene una manera propia de funcionar, un criterio de clasificación. Para un mismo conjunto de entrenamiento dos algoritmos de clasificación pueden producir clasificadores distintos. Muchos investigadores han explotado este hecho para crear sistemas de múltiples clasificadores donde los clasificadores individuales sean distintos.

La eficacia de los algoritmos de clasificación puede variar según la región del espacio de representación, puede que para un problema dado un algoritmo de clasificación sea mejor que otro en solo una zona del espacio de representación. En [52] se propone utilizar 4 tipos distintos de algoritmos de clasificación, después estimar la eficacia de cada uno de ellos en distintas regiones del espacio de

representación. En la etapa de clasificación se calcula de manera dinámica el clasificador más eficaz en la región del objeto en cuestión y se selecciona este para la clasificación.

En [86] se muestra cómo al introducir unos pocos árboles de decisión en esquemas de redes neuronales aumenta la diversidad del sistema de múltiples clasificadores y por lo tanto también la eficacia. También se muestra que cuando el número de árboles de decisión aumenta la eficacia de la combinación se deteriora. Creemos que el problema de estos resultados es que son obtenidos de una forma empírica, es decir, en uno o varios problemas particulares y no presentan una base teórica detrás.

La eficacia de los clasificadores que produce un algoritmo de clasificación depende en gran medida de la correspondencia entre el criterio que este utiliza para clasificar y el problema real en cuestión. A veces la realidad es tan compleja que para ajustarse a ella de manera correcta se necesitan varios criterios. Por ejemplo la Figura 3 se muestra una distribución de objetos que pueden pertenecer a una de 3 clases —verde, rojo y amarillo–, se puede notar como un clasificador de discriminante lineal puede discriminar bien a los objetos de la clase amarillo, pero no ocurre lo mismo con las otras clases. Quizás sería conveniente utilizar otro tipo de algoritmo en la zona de las clases verde y rojo.

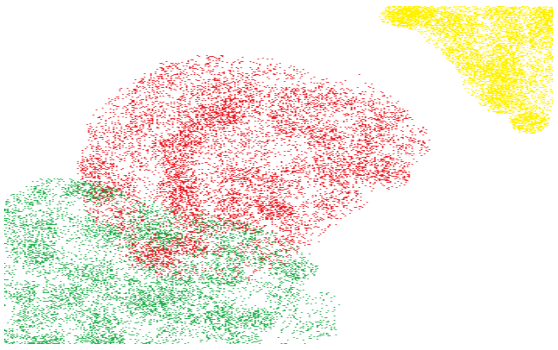

Fig. 3. Un espacio de representación de 2 dimensiones con objetos de 3 clases: verde, rojo y amarillo.

4.3 División del problema general

Divide y vencerás ha sido probablemente la frase más repetida –y agradecida– en toda la historia de la ciencia de la computación. La idea del mensaje es que en muchas ocasiones puede ser más fácil resolver varios problemas simples y después combinar los resultados que resolver el problema complejo directamente. El campo de la combinación de clasificadores no ha resistido la tentación. Los aportes más serios en este sentido van en la dirección de atacar el problema de la clasificación con múltiples clases con algoritmos de clasificación desarrollados para resolver problemas con solo 2 clases, para lograr esto existen dos enfoques principales:

- Uno contra el resto: este enfoque divide un problema de clasificación con $k > 2$ clases en k problemas de clasificación con 2 clases, cada uno de estos problemas consiste en saber si el objeto pertenece a la clase K_i, que ahora sería K_1, o al resto $\Omega - K_i$, que ahora sería K_2.
- Uno contra uno: este enfoque divide un problema de clasificación con $k > 2$ clases en $k(k+1)/2$ problemas de clasificación con 2 clases e intenta discriminar entre todos los pares de clases.

Por ejemplo, un algoritmo que trabaja con el primer enfoque es el presentado en [87] y otro que utiliza el segundo enfoque es el Adaboost.M2, los dos son variantes de Boosting. Si se desea profundizar en el segundo enfoque desde un punto de vista general el lector puede remitirse a [88].

A continuación se expondrá uno de los métodos más utilizados en la combinación de clasificadores y que permite modelar los enfoques anteriores.

4.3.1 Error Correcting Output Codes (ECOC)

ECOC [89] propone la idea de atacar los problemas de clasificación con múltiples clases a partir de una división de los mismos en sub-problemas que clasificación de 2 clases. De esta forma cada clasificador en la combinación discrimina entre 2 nuevas clases que son creadas a partir de las originales. Por ejemplo, para un problema con 6 clases el clasificador D_i podría discriminar entre las clases $K'^i_1 = \{K_1, K_3\}$ y $K'^i_2 = \{K_2, K_4, K_5, K_6\}$. Nótese que la notación que se utilizará en lo adelante K'^i_1 y K'^i_2 denotarán la 2 clases presentes en el sub-problema que resuelve el clasificador D_i.

Nótese que cada sub-problema puede ser codificado por una tupla de ceros y unos de k dimensiones, para el ejemplo anterior dicha tupla sería $(1, 0, 1, 0, 0, 0)$. Formalmente la tupla que representa el sub-problema que el clasificador D_j debe resolver, tiene un 1 en la posición i si $K_i \in K'^j_1$ y 0 en otro caso.

Los l sub-problemas que *ECOC* tiene que resolver se codifican entonces mediante una matriz de k filas por l columnas, a dicha matriz se le llamará matriz de códigos (*code matrix* en inglés) y será denotada por C. La entrada (i, j), con $1 \leq i \leq k$ y $1 \leq j \leq l$ de la matriz C tendrá un 1 si $K_i \in K'^j_1$ y 0 en otro caso. Denotaremos por w_i a la tupla que codifica la fila número i de la matriz C, a estas tuplas se les conoce en inglés como *codewords*.

Entonces una vez que se tiene la matriz de códigos que se va a utilizar se entrenan l clasificadores con l algoritmos de clasificación no necesariamente distintos, cada uno en su respectivo sub-problema. Una vez entrenados los clasificadores, en la etapa de clasificación se codifica la salida de los clasificadores como una tupla de l dimensiones que denotaremos por d y que tiene un 1 en la posición número i si el clasificador D_i asigna el objeto en cuestión a la clase K'^i_1. El método *ECOC* da como resultado la clase K_j si de todas las *codewords* w_j es la que tiene menor distancia de Hamming con d.

Existen distintos enfoques para obtener un matriz de códigos:

- Un código por clase: este enfoque permite modelar una división al estilo uno contra el resto, en este caso se utilizan $l = k$ clasificadores y el *codeword* w_i tiene un 1 en la posición número i y 0 en el resto de las posiciones.
- Todos los posibles códigos: este enfoque modela todas las posibles divisiones del problema, la matriz de códigos resultantes tiene k filas y $2^{(k-1)} - 1$ columnas.
- Códigos aleatorios: en este enfoque se genera la matriz de códigos de manera aleatoria.

En [9] se plantea que el método ECOC reduce el sesgo y la varianza del algoritmo de clasificación utilizado. Según dichos autores el hecho de los clasificadores se entrenen en distintos sub-problemas reduce el sesgo del algoritmo, y si el algoritmo de clasificación es inestable entonces también se reduce la varianza. Los autores de [9] creen que es conveniente utilizar el método ECOC con algoritmos de clasificación que tengan un "enfoque global" para clasificar, o sea, que para clasificar un nuevo objeto se tenga en cuenta información relacionada con todo el espacio de representación, como los basados en redes neuronales y árboles de decisión. Por otra parte, algoritmos de clasificación que tiene un carácter muy local como los basados en los vecinos más cercanos no deben beneficiarse mucho del método. A estas conclusiones ellos llegan a través de la vía experimental pero es algo sobre lo que creemos que se debe profundizar más.

Es importante notar que el método *ECOC* es impracticable en problemas con muchas clases debido a la cantidad de clasificadores a entrenar.

5 Diversidad

Los autores en [90] señalan que el concepto de diversidad es sin duda el más utilizado en el campo de la combinación de clasificadores, tanto por su utilidad teórica, ya que permite estudiar y analizar el comportamiento de los métodos de combinación de clasificadores, como por su utilidad práctica, ya que permite desarrollar nuevos algoritmos. Sin embargo también es el concepto menos formalizado, debido

a esto no existe homogeneidad ni en el sentido, ni en la utilización que se le da. Según dichos autores, en la comunidad hay consenso sobre las ideas de que:

- Existe una propiedad de las combinaciones de clasificadores que puede ser llamada *diversidad*, esta puede ser definida cuantitativamente y por lo tanto medida, y se relaciona de una manera estrecha con la eficacia de la combinación.
- Dicha propiedad puede ser utilizada para construir combinaciones de clasificadores eficaces.

Sin embargo, y como bien se apunta en [90], dentro de la comunidad también han surgido opiniones que cuestionan la utilidad práctica del estudio de un concepto como la diversidad. Entre los trabajos que plantean esta disyuntiva se encuentran [91], [1] y [32].

Está claro que si se tiene un clasificador perfecto que no comete errores entonces no es necesario utilizar una combinación de clasificadores para resolver un problema. Pero si todos los clasificadores individuales de una combinación cometen los mismos errores entonces tampoco vale la pena utilizar la combinación.

La diversidad en la combinación de clasificadores ha sido estudiada desde distintos aspectos: se han propuesto medidas para cuantificar la diversidad entre un par –o un grupo– de clasificadores, se han propuesto descomposiciones del error de la combinación tratando de obtener relación con alguna magnitud que pueda ser interpretada como diversidad, se han propuesto métodos de generación de clasificadores individuales de forma tal que se optimice una noción de diversidad dada. A continuación se abunda sobre estas y otras cuestiones.

5.1 Medidas de diversidad

Que el concepto de diversidad en la combinación de clasificadores aún no se haya definido correctamente no ha sido impedimento para el desarrollo de varias medidas de esta. Ya Sharkey en [92] propone una medida cualitativa de la diversidad en esquemas de redes neuronales, aunque dicha medida puede ser tenida en cuenta con cualquier tipo de combinación. La medida define 4 niveles de diversidad:

1- Los clasificadores no cometen errores coincidentes. En este caso el voto mayoritario siempre da un resultado correcto.
2- Se cometen errores coincidentes pero la mayoría siempre da un resultado correcto. El voto mayoritario también da siempre un resultado correcto.
3- A veces la mayoría se equivoca, pero siempre hay al menos un clasificador individual que realiza una clasificación correcta. El voto mayoritario no siempre da un resultado correcto.
4- A veces ocurre que todos los clasificadores se equivocan.

El nivel de diversidad de una combinación se asigna según el peor caso registrado. Según nuestra opinión, esta constituye la principal desventaja de esta medida, ya que el nivel de la diversidad de una combinación no da una noción acerca de cómo se comporta esta en todo un conjunto de objetos. Por ejemplo, se puede tener una combinación donde todos los clasificadores individuales den un resultado correcto para todos los objetos excepto para uno en el cual se equivocan todos. Dicha combinación tendrá un cuarto nivel de diversidad cuando en casi todo el conjunto de objetos la diversidad se comporta al primer nivel.

Una propuesta de solución al problema anterior es dada en [93], esta consiste en asignarle un nivel de diversidad a cada objeto en el conjunto. De esta forma se cuentan cuantos objetos la combinación coloca en cada nivel, así por ejemplo se pueden tener 50 objetos en el primer nivel y 1 en el cuarto o tener 1 en el primer nivel y 50 en el cuarto, aunque en ambos casos la combinación tiene una diversidad en el cuarto nivel se sabe que el primero es mucho mejor que el segundo.

Las medidas cuantitativas propuestas en la literatura de la combinación de clasificadores son usualmente divididas en dos categorías: las que miden la diversidad entre un par de clasificadores y las que miden la diversidad en un grupo de clasificadores. Si se desea medir la diversidad de un grupo de

clasificadores usando una medida entre pares, se puede calcular la diversidad entre todos los $(l(l-1))/2$ pares y después promediar el resultado.

Para calcular la diversidad entre un par de clasificadores D_i y D_j en un conjunto de objetos V usualmente se construye una tabla (Tabla 9) que codifica las salidas de los clasificadores. Por ejemple en dicha tabla el valor a se corresponde con el número de objetos en V que ambos clasificadores asignaron a la clase correcta. Los valores de la Tabla 9 serán utilizados en la definición de medidas de diversidad en lo adelante. Nótese que $(a+b+c+d) = |V|$.

Tabla 9. Codificación de los resultados de los clasificadores D_i y D_j.

	D_i Correcto.	D_i Incorrecto.
D_j Correcto.	a	b
D_j Incorrecto.	c	d

Es importante notar que las medidas que utilicen los valores a, b, c, d, solo están tomando información abstracta de la salida de los clasificadores.

5.1.1 Medida de desacuerdo

Es una medida entre pares, y probablemente la más intuitiva de todas las medidas ya que simplemente cuenta los desacuerdos –cuando uno se equivoca y el otro no– entre el par de clasificadores. Así la formula quedaría:

$$dis_{i,j} = \frac{b+c}{|V|}. \tag{28}$$

El valor de la medida está en el intervalo [0,1], a mayor $dis_{i,j}$ mayor desacuerdo y por tanto más diversidad. Sin ser llamada medida de desacuerdo esta medida ha sido usada en [65] y [94]. Una característica de esta medida es que trata de la misma forma el hecho de que los clasificadores estén de acuerdo correctamente que incorrectamente.

5.1.2 La estadística Q

Originalmente propuesta por [95] la estadística Q también puede ser utilizada para medir la diversidad entre dos clasificadores:

$$Q_{i,j} = \frac{ad-bc}{ad+bc}. \tag{29}$$

El valor de la medida está en el intervalo $[-1,1]$. Dos clasificadores estadísticamente independientes tendrán un valor cercano a 0, en cambio, si tienden a clasificar correctamente a los mismos objetos tendrán un valor cercano a 1, y si tienden a cometer errores en distintos objetos tendrán un valor cercano a -1.

5.1.3 Coeficiente de correlación

El coeficiente de correlación [96] se expresa de la siguiente manera:

$$\rho_{i,j} = \frac{ad-bc}{\sqrt{(a+b)(a+c)(c+d)(b+d)}}. \tag{30}$$

Esta medida es muy similar a la estadística Q, de hecho ambas siempre tienen el mismo signo y además se verifica que $|\rho_{i,j}| \geq |Q_{i,j}|$.

5.1.4 Basada en la varianza según Kohavi y Wolpert

Esta no es una medida de diversidad definida entre pares sino entre un grupo de clasificadores. En [97] propone la siguiente medida de la varianza de un algoritmo de clasificación A en un punto x:

$$var(A,Z,x) = E_Z[\frac{1}{2}\left(1 - \sum_{i=1}^{k} P(A(Z,x) = K_i|x)^2\right)].$$

En la ecuación anterior E_Z representa la esperanza matemática sobre distintos conjuntos de entrenamiento.

En [1] y [32] se utiliza esta idea para de obtener una medida de diversidad como se verá a continuación. Se codificará la salida de los clasificadores en $\{1, 0\}$, o sea, correcta e incorrecta. En el caso de la medida de Kohavi y Wolpert la estimación es a través de distintos conjuntos de entrenamiento, para la medida de diversidad es a través de los distintos clasificadores. O sea denotaremos por $P(1|x) = \frac{\sum_{i=1}^{l} D_i(x)}{l}$, y $P(0|x) = \frac{l - \sum_{i=1}^{l} D_i(x)}{l}$, sustituyendo en la fórmula de Kohavi se tiene:

$$KW(x) = \frac{1}{2}(1 - P(1|x)^2 - P(0|x)^2).$$

Promediando para todos los objetos en el conjunto V y después de varias transformaciones algebraicas se obtiene la medida

$$KW(V) = \frac{1}{|V|l^2}\left[\sum_{x \in V}\left(\sum_{i=1}^{l} D_i(x)\left(l - \sum_{i=1}^{l} D_i(x)\right)\right)\right]. \qquad (31)$$

Una muestra mucho más extensa de medidas de diversidad puede ser encontrada en [1], [32], [98] y [99]. Es importante destacar que la mayoría de las medidas están correlacionadas entre si y que hasta el momento ninguna se ha podido relacionar directamente ni al error de la combinación ni a la eficacia.

5.2 Descomposiciones del error de clasificación de la combinación

Otra de las direcciones seguidas en el tema de la diversidad es la descomposición del error de clasificación de la combinación en función de algún término que pueda ser interpretado como la diversidad entre los clasificadores individuales. Por supuesto, dicha descomposición dependerá de la función de pérdida –o costo del error– y de la función de combinación que se utilice. Los esfuerzos han estado dirigidos a obtener una descomposición utilizando la función de pérdida en ceros y unos y el voto mayoritario.

Las investigaciones en este sentido están inspiradas en los excelentes resultados alcanzados en el caso de la combinación de estimadores reales –regresión– utilizando el promedio como función de combinación y el cuadrado de la diferencia como función de pérdida. Como bien se observó al inicio de este trabajo la esperanza del error de un estimador de números reales se puede descomponer en los términos de sesgo y varianza [6]. Sucede entonces que cuando se combinan varios estimadores reales se puede descomponer el error de dicha combinación en términos de sesgo, varianza y covarianza, donde se muestra a las claras que al disminuir la correlación entre los estimadores disminuye el costo del error.

Sea un problema de regresión con los $h_i(Z,x) \approx f(x)$, $1 \leq i \leq l$ como los estimadores individuales de una combinación de estimadores reales $h(Z,x) = \frac{1}{l}(\sum_{i=1}^{l} h_i(Z,x))$. En [100] se muestra que

$$E\left[\left(h(Z,x) - f(x)\right)^2\right] = \overline{sesgo}^2 + \frac{1}{l}\overline{varianza} + (1 - \frac{1}{l})\overline{covarianza}. \qquad (32)$$

En la ecuación anterior se tiene que:

$\overline{sesgo} = \frac{1}{l}\sum_{i=1}^{l} E[h_i(Z,x) - f(x)]$,

$\overline{varianza} = \frac{1}{l}\sum_{i=1}^{l} E[(h_i(Z,x) - E[h_i(Z,x)])^2]$,

$\overline{covarianza} = \frac{1}{l(l-1)}\sum_{i=1}^{l}(\sum_{j=1,i\neq j}^{l} E[(h_i(Z,x) - E[h_i(Z,x)])(h_j(Z,x) - E[h_j(Z,x)])])$.

O sea el primer término es el promedio de los sesgos de los estimadores individuales, es segundo el promedio de las varianzas de los estimadores individuales, y el tercero es el promedio de las covarianzas ente los estimadores individuales. Lo ideal aquí sería disminuir la covarianza entre los estimadores individuales sin aumentar el sesgo ni la varianza.

Otro enfoque seguido en el caso de la regresión es descomponer el error en un término usualmente conocido como ambigüedad y que puede ser interpretado como la diversidad entre los clasificadores [93], [90]. Dicha descomposición fue mostrada en [101] y prueba que en un punto x del espacio de representación el cuadrado de la diferencia entre la función real a aproximar f y la combinación h de un conjunto de estimadores individuales h_i es inferior o igual al promedio de los cuadrados de las diferencias entre los estimadores individuales con f. La descomposición es la siguiente:

$$(h(Z,x) - f(x))^2 = \frac{1}{l}\sum_{i=1}^{l}(h_i(Z,x) - f(x))^2 - \frac{1}{l}\sum_{i=1}^{l}(h_i(Z,x) - h(Z,x))^2. \qquad (33)$$

El segundo término de la descomposición es el llamado término de la ambigüedad, nótese que es el promedio del cuadrado de las diferencias del resultado de la combinación con cada uno de los resultados de los estimadores individuales y que por tanto siempre es mayor o igual a cero. A mayor ambigüedad mayor reducción del error, pero nótese que según aumenta la variabilidad de los estimadores individuales también aumenta el valor del primer término, lo que refleja que la diversidad por sí sola no es de gran utilidad ya que es necesario un balance entre esta y la precisión de los estimadores reales.

La descomposición en el término de ambigüedad tiene un gran tiene un gran poder, pues no tiene en cuenta esperanzas matemáticas sobre conjuntos de entrenamientos, además es práctica y expresiva. Por estas razones ha sido utilizada en la construcción de combinaciones eficaces de estimadores reales, por ejemplo el método propuesto en [102] y fundamentado en [103].

En el área de clasificación –cuando el resultado del estimador es una clase– no se ha podido obtener una descomposición similar a la (32) lo que no es de extrañar pues como los resultados de los clasificadores no son valores numéricos no se puede definir la covarianza entre estos. En [23] se propone una descomposición similar a la (33) para el caso en que se cuenta solamente con dos clases.

Considérese un grupo de clasificadores individuales D_i, con $1 \leq i \leq l$ y l impar, para lograr una mayor facilidad y entendimiento en la notación, las clases serán representadas por los valores $\{+1, -1\}$ y se asume que los clasificadores dan una salida abstracta, o sea $D_i(x) \in \{+1, -1\}$. Representemos por $y(x)$ la clase a la cual pertenece el objeto representado por x, el costo en que incurre el clasificador D_i al clasificar el objeto x será representado por la función de costo de error en ceros y unos:

$$e_i(x) = \frac{1}{2}(1 - y(x)D_i(x)).$$

Nótese que $e_i(x)$ es igual a 0 si el clasificador D_i clasifica correctamente al objeto representado por x y 1 en otro caso. La combinación de los clasificadores individuales por voto mayoritario será representada por $D(x) = sign(\frac{1}{l}\sum_{i=1}^{l} D_i(x))$ donde $sign(x)$ es la función signo y da como resultado $+1$ si $x > 0$ y -1 si $x < 0$ (recuérdese que se asume que l es impar por lo que $D(x)$ siempre es distinto de 0). El error de la combinación será representado por:

$$e(x) = \frac{1}{2}(1 - y(x)D(x)).$$

Se define además el desacuerdo entre el clasificador D_i y la combinación D como:

$$\delta_i(x) = \frac{1}{2}(1 - D_i(x)D(x)).$$

Después de varias manipulaciones a partir de la diferencia entre el error de la combinación y el promedio de los errores individuales se llega a la siguiente formula:

$$e(x) = \frac{1}{l}\sum_{i=1}^{l} e_i(x) - y(x)D(x)\frac{1}{l}\sum_{i=1}^{l}\delta_i(x). \qquad (34)$$

Esta ecuación demuestra que para el caso de dos clases la diferencia entre el costo del error de la combinación mediante voto mayoritario y el promedio de los costos de los errores individuales puede ser expresado en términos del desacuerdo entre los clasificadores individuales. Ahora, nótese lo siguiente, a diferencia de la relación obtenida en la ecuación (33) el término de ambigüedad en este caso incluye la clase verdadera a la cual pertenece el objeto representado por x. Es fácil notar que el papel que juega el término de la ambigüedad en este caso está afectado por el valor de $y(x)D(x)$, o sea por si la combinación clasifica correctamente o no al objeto en cuestión.

Nótese que si la combinación clasifica correctamente al objeto entonces el desacuerdo entre los clasificadores individuales es beneficioso, en cambio si la combinación clasifica incorrectamente al objeto el descuerdo entre los clasificadores individuales es perjudicial. En [23] se relaciona este resultado al patrón de éxito y patrón de fracaso ambos propuestos por Kuncheva y ya estudiados en este reporte.

En [90] se propone una posible descomposición parecida a la mostrada en (32) para el caso de la clasificación supervisada, basada en la descomposición de sesgo y varianza propuesta en [7] –que además fue mostrada al inicio de este reporte–, pero esta es extremadamente compleja y se encuentra todavía bajo el estudio de sus autores. Además se propone una generalización de la descomposición de la (34) para el caso en que se tienen más de dos clases, lo que en este caso para los desacuerdos se tiene en cuenta si el resultado de los clasificadores es correcto o no.

6 Categorizaciones

En la literatura han sido propuestos varios métodos para caracterizar las combinaciones de clasificadores. Por ejemplo Sharkey [104] propone una taxonomía para los esquemas de redes neuronales, pero que en realidad se puede aplicar a todos los esquemas y combinaciones en general, que contempla las siguientes tres dimensiones:

1- Si los clasificadores en la base son competitivos o cooperativos, esto es, cuando los clasificadores son competitivos solo se toma en cuenta el resultado de uno de ellos: el que supuestamente ganó la competencia, en cambio cuando son cooperativos se combinan todas las decisiones.
2- Si el esquema se define de arriba hacia abajo (*top-down*) o si se hace de abajo hacia arriba (*bottom-up*), en el primer caso se encuentran los sistemas que no tienen en cuenta la salida de los clasificadores en la combinación de estos, en el segundo caso los que sí lo hacen. Los métodos cooperativos son por definición *bottom-up* ya que para combinar los resultados tienen que necesariamente tenerlos en cuenta. Por otra parte, la selección de los clasificadores suele ser *top-down* ya que no es usual analizar las salidas de estos, aunque también puede ocurrir que las salidas sean analizadas.
3- Si se combinan clasificadores que resuelven una misma tarea, clasificadores que resuelven una parte más pequeña del problema original, o simplemente un híbrido entre los dos casos anteriores.

Ho [105], [106] divide los esquemas en dos tipos (Valentini y Masulli [107] manejan los mismos conceptos pero utilizando los términos de combinaciones generativas y no generativas):

1- Los que optimizan la decisión, estos métodos se abstraen de cómo son generados los clasificadores individuales y se centran en combinar los resultados de estos de una manera óptima.
2- Los que optimizan la generación de los clasificadores individuales, o sea, cómo va a ser entrenado cada clasificador individual.

En [1] se señala que al existir varios tipos de esquemas que crean los clasificadores individuales y a su vez definen una manera de combinar los resultados de estos (Adaboost), no es posible clasificar todos los tipos de combinaciones de clasificadores existentes en uno de los dos tipos propuestos por Ho. Por otra parte Kuncheva [1] propone agrupar las combinaciones según la forma en que estas son creadas, para esto se definen cuatro capas de trabajo:

1- Capa de combinación. Define la forma en que los clasificadores se combinan.
2- Capa de clasificadores. Define qué clasificadores utiliza la combinación.
3- Capa de rasgos. Define qué rasgos son usados por cada clasificador.
4- Capa de los datos. Define con qué datos se entrena cada clasificador.

No obstante Kuncheva reconoce que el modelo anterior tampoco contempla todas las combinaciones posibles, por ejemplo, según ella, ECOC [89] no puede ser tenido en cuenta a través de dichas capas de trabajo. Por lo tanto no puede ser considerado una taxonomía de las combinaciones de clasificadores.

Duin [108] propone diferenciar las combinaciones de clasificadores en las que reciben un entrenamiento aun cuando los clasificadores individuales ya fueron entrenados, y las que no. En este sentido se pueden señalar tres categorías: las que no necesitan un entrenamiento adicional, por ejemplo voto mayoritario, las que necesitan dicho entrenamiento, por ejemplo BKS, y las que van desarrollando su función de combinación durante el entrenamiento de los clasificadores individuales, por ejemplo Adaboost. Otra terminología manejada en la literatura [109] es la de combinación dependiente o independiente de los datos, ya que usualmente el entrenamiento se realiza sobre los datos. Es importante tener en cuenta que las combinaciones que dependen de los datos, o sea, las que requieren un entrenamiento adicional pudieran asumir condiciones acerca de estos que en realidad no se cumplen para el problema en cuestión. Otros aspectos a tener en cuenta son la complejidad y el peligro de sobreajuste, ya que es muy usual que los que necesitan entrenamiento sean los más complejos y más dados al sobreajuste.

Gavin Brown [93] propone dividir las combinaciones de clasificadores en las que crean los clasificadores individuales tratando de optimizar una métrica de diversidad de forma explícita y las que no, o sea en estas últimos la diversidad se obtiene de forma implícita, por ejemplo a través de la aleatoriedad.

En [14] Rokach propone una nueva taxonomía que intenta categorizar los tipos de combinaciones más significativos en la literatura y la práctica hasta el momento, el objetivo es ayudar a distinguir entre los esquemas ya existentes y a identificar y explorar nuevas combinaciones poco exploradas. Además propone varios criterios para una mejor selección de la combinación adecuada a un problema dado. Para su taxonomía Rokach identifica las siguientes componentes indispensables en la tarea de la clasificación automática a través de la combinación de clasificadores: el conjunto de entrenamiento, el algoritmo de clasificación, el generador del esquema, y la función de combinación; y entonces categoriza las combinaciones de clasificadores utilizando la naturaleza de estas componentes y además las relaciones entre estas. La taxonomía resultante cuenta con las siguientes dimensiones:

1- ¿Qué uso se le da a la función de combinación? Esta dimensión describe la relación entre el generador del esquema y la función de clasificación.
2- La dependencia entre los clasificadores. ¿Se afectan entre sí los clasificadores durante el entrenamiento? Esta dimensión dice si los clasificadores son dependientes o independientes.
3- ¿Cómo se obtiene la diversidad? En esta dimensión se dice cómo se intenta conseguir la diversidad entre los clasificadores. Por ejemplo, a través de distintos conjuntos de entrenamiento, a través de variaciones en el algoritmo de clasificación, etc.

4- El tamaño del esquema. Cuántos clasificadores hay en el esquema y cómo se eliminan los no deseados.
5- Soporte a los algoritmos de clasificación. Algunos esquemas son desarrollados para algoritmos de clasificación específicos, o al menos algoritmos que tengan ciertas propiedades, por ejemplo, Random forest fue creado para trabajar con algoritmos de clasificación sobre árboles de decisión, otros por el contrario son independientes del algoritmo que se use para entrenar.

A pesar de los diversos esfuerzos por encontrar una taxonomía unificada y útil, hay que decir que hasta el momento no existe ninguna a la que se le considere perfecta, o más aceptada; nótese que los autores de todas las taxonomías consultadas reconocen que existen métodos no contemplados por su propuesta. Se ha llegado incluso a cuestionar la necesidad de una taxonomía para el campo.

7 Conclusiones

Es importante tener en cuenta numerosas cuestiones a la hora de aplicar una combinación de clasificadores en la práctica, ya que el hecho de que un método en particular tenga éxito en un contexto o problema dado no garantiza el éxito en otros contextos o problemas. Creemos que lo primero a tener en cuenta es que el modelo que estemos usando no suponga nada que no se verifique en la práctica.

Por ejemplo, cuando usamos funciones de combinación de salidas es importante tener en cuenta el tipo de las salidas de cada uno de los clasificadores individuales. Además, cuando todas las salidas sean de tipo medida, hay que tener en cuenta que dichas salidas sean homogéneas ya que los clasificadores individuales pudieran dar distintos tipos de soportes. Esto es importante ya que las funciones de combinación están diseñadas para trabajar con tipos de salidas en específico, y en el caso de las salidas de tipo medida, muchas funciones de combinación asumen que los soportes son de la misma naturaleza, el ejemplo más claro son las funciones máximo y mínimo.

Respecto a las funciones de combinación, otros aspectos a tener en cuenta son:

- La cantidad de clases y de clasificadores individuales con los que se cuentan: si las cantidades anteriores son grandes quizás no convenga utilizar una función de combinación de las del tipo multinomial.
- La cantidad de objetos disponibles para el entrenamiento: si no se tienen suficientes objetos para realizar el entrenamiento no se tiene otra opción que utilizar métodos simples como el voto mayoritario, pero si la muestra para entrenar es abundante se pueden considerar otras opciones.
- El desempeño de los clasificadores individuales: si los clasificadores individuales tienen una eficacia parecida los métodos simples son preferidos, en cambio si la eficacia de estos clasificadores es distinta quizás métodos complejos den mejores soluciones.

Un asunto fundamental a tener en cuenta son los algoritmos de clasificación que se utilizan en la creación de los clasificadores individuales. Hay que garantizar que el modelo matemático que soporta a dichos algoritmos se ajuste al problema en cuestión.

También es importante tener en cuenta la estrategia de generación que se vaya usar, por ejemplo, si se va a utilizar Boosting no es conveniente que los clasificadores individuales resultantes sean muy complejos.

La cantidad de objetos en el conjunto de entrenamiento también es una cuestión a tener en cuenta aquí, por ejemplo, si se tiene una muestra pequeña y a esta se le aplica Bagging, los clasificadores individuales serán entrenados con muy pocos objetos.

La cantidad de dimensiones del espacio de representación del problema también debe ser un factor a tener en cuenta, resulta claro que si se tienen pocas dimensiones un algoritmo como *Random forest* no tiene por qué dar buenos resultados.

El problema de la diversidad entre los clasificadores individuales sigue siendo el centro de una gran cantidad de opiniones encontradas. El debate principal gira entorno a si esta puede ser explotada en la creación de mejores combinaciones o no. Su estudio se mantiene activo, aunque los resultados obtenidos hasta el momento son poco alentadores.

El campo de la combinación de clasificadores es muy diverso. Existen muchos algoritmos disponibles para ser utilizados, sin embargo, muchos de estos aun no tienen una explicación fundamentada acerca del porqué de su funcionamiento, incluso cuando los mismos han sido utilizados con éxito en la práctica (por ejemplo *Random forest*). Por otra parte, existen metodologías con una fundamentación teórica muy fuerte (por ejemplo *Boosting*).

Son tantos los puntos de vistas desde los cuales puede ser analizado el campo de la combinación de clasificadores, que aún no existe una taxonomía completamente coherente para el mismo, a pesar de los intentos de varios de los científicos que lideran el campo.

No obstante el éxito obtenido por los métodos de combinación de clasificadores, estos deben ser considerados solamente como una herramienta más, la cual no tiene por qué ser adecuada para todos los problemas. Es decir, siempre que sea posible resolver un problema de manera satisfactoria con un clasificador individual, no debe utilizarse ningún método de combinación.

8 Investigaciones futuras

En este trabajo se ha reiterado varias veces la necesidad de dar una fundamentación teórica a varios métodos utilizados en la combinación de clasificadores, pues, aunque en muchos casos estos se aplican de manera satisfactoria en la práctica, el desconocimiento sobre el funcionamiento de dichos métodos impide tanto el desarrollo –o mejoramiento– de los mismos como la justificación de su mal funcionamiento en ciertos casos o incluso sus resultados positivos.

La misma Kuncheva expresa en [1] que pareciera que los científicos han estado más preocupados creando nuevos métodos que estudiando o justificando los ya existentes. En este sentido, una de las posibles direcciones a seguir en futuras investigaciones es dar justificaciones correctas acerca del funcionamiento de estos métodos. A continuación presentamos una selección de los métodos –o ideas– que creemos deben ser justificados de una manera más formal en el campo de la combinación de clasificadores:

- En los últimos años se han venido proponiendo distintos modelos para demostrar la convergencia del método *Random forest* –véase el epígrafe de *Random forest*–, sin embargo, aun el algoritmo más general no ha sido atacado directamente, y en su lugar han sido analizados modelos más simples, o sea, versiones simplificadas del mismo. Además, todavía falta por explicar la eficacia del mismo de manera rigurosa.
- En los métodos de selección de clasificadores, en muchos casos se estima la competencia de cada clasificador a partir de cierta medida de su eficacia local, o sea, su eficacia en una vecindad del objeto a clasificar. En estos casos se asume que el espacio de representación es un espacio métrico. Sería conveniente analizar en detalle las implicaciones de utilizar una función de similitud en los casos en los que el espacio de representación no sea un espacio métrico.
- El voto mayoritario es sin dudas la función de combinación de salidas más utilizada en todo el campo de la combinación de clasificadores. La teoría sobre el voto mayoritario y la votación en general es bien amplia, sin embargo, aún no se ha podido dar una expresión de la eficacia de este método de combinación en función de la complementariedad de los clasificadores individuales. Nótese que también es necesario formalizar el concepto de complementariedad entre los clasificadores individuales.

Otra posible dirección para futuras investigaciones es el enfoque de la poda de los clasificadores individuales. En este enfoque primeramente se producen muchos clasificadores individuales y después se intenta extraer un subconjunto más pequeño de estos clasificadores de forma tal que cuando estos

sean combinados se obtengan buenos resultados. Por ejemplo, la selección de este subconjunto de clasificadores pudiera hacerse teniendo en cuenta la eficacia de los mismos, o quizás teniendo en cuenta alguna medida de diversidad entre los clasificadores individuales resultantes. Sería importante analizar si seleccionar los clasificadores individuales a partir de la diversidad entre estos provee alguna ventaja sobre la selección basada directamente en la eficacia.

Las descomposiciones del error de clasificación en términos que puedan representar la diversidad entre los clasificadores –ver epígrafe de diversidad– también merecen atención para futuras investigaciones. Sería interesante continuar analizando la utilidad de las descomposiciones propuestas en [110] y [111] o proponer nuevas descomposiciones. Es importante notar en este punto, que la descomposición del error dependerá necesariamente de la función de combinación de salidas que se utilice. La descomposición más estudiada hasta el momento es a partir del voto mayoritario, pero también pudieran ser utilizadas otras funciones de combinación.

También es recomendable investigar sobre las aplicaciones que tienen –y pueden tener– estos métodos en la Biometría.

Referencias bibliográficas

1. L. I. Kuncheva, Combining Pattern Classifiers Methods and Algorithms, John Wiley & Sons, 2004.
2. G. S. Sebestyen, «Decision-Making Process in Pattern Recognition,» The Macmillan Company, New York, 1962.
3. L. Rokach, Pattern classification using ensemble methods, vol. 75, World Scientific, 2010.
4. J. W. Tukey, Exploratory data analysis, Addison-Wesley, 1977.
5. L. Xu, A. Krzyzak y C. Y. Suen, «Methods of combining multiple classifier and their application to handwriting recognition.,» *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, nº 3, pp. 418-435, 1992.
6. S. Geman, E. Bienenstock y R. Doursat, «Neural Networks and the Bias/Variance Dilemma,» *Neural Computation*, vol. 4, nº 1, pp. 1-58, 1992.
7. R. Kohavi y D. H. Wolpert, «Bias Plus Variance Decomposition for Zero One Loss Functions,» de *Machine Learning Proceedings of the 13th International Conference*, 1996.
8. J. V. Hansen, «Combining predictors: Meta machine learning methods and bias/variance & ambiguity decompositions,» Ph.D. Dissertation, 2000.
9. E. B. Kong y T. .. G. Dietterich, «Error-correcting output coding corrects bias and variance,» de *Proceedings of the Twelfth International Conference on Machine Learning*, Tahoe City, 1995.
10. L. Breiman, «Bias, variance and arcing classifiers,» Tech. Rep. 460, Statistics Department, University of California, Berkeley, CA, USA, Berkeley, 1996.
11. R. Tibshirani, «Bias, variance and prediction error for classification rules,» Department of Preventive Medicine and Biostatistics and Department of Statistics, University of Toronto, Toronto, 1996.
12. J. H. Friedman, «On bias, variance, 0/1 - loss, and the curse-of-dimensionality,» *Data mining and knowledge discovery*, vol. 1, nº 1, pp. 55-77, 1997.
13. P. Domingos y G. Hulten, «A unified bias-variance decomposition and its applications,» de *Proceedings of the 17th International Conference on Machine Learning*, 2000.
14. L. Rokach, «Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography.,» *Computational Statistics & Data Analysis*, vol. 53, nº 12, p. 4046–4072, 2009.
15. T. G. Dietterich, «Machine-learning research: Four Current Directions,» *AI magazine*, vol. 18, nº 4, p. 97, 1997.
16. T. G. Dietterich, «Ensemble methods in machine learning,» de *Multiple classifier systems*, Cagliari, Springer, 2000, pp. 1-15.
17. L. Hyafil y R. L. Rivest, «Constructing optimal binary decision trees is NP-complete,» *Information Processing Letters*, vol. 5, nº 1, pp. 15-17, 1976.

18. A. Blum y R. L. Rivest, «Training a 3-node neural network is NP-complete (Extended Abstract),» de *Workshop on Computational Learning Theory*, San Francisco, 1988.
19. S. Site y S. K. Mishra, «A Review of Ensemble Technique for Improving Majority Voting for Classifier,» *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, nº 1, 2013.
20. H. V. Georgiou y M. E. Mavroforakis, «A game-theoretic framework for classifier ensembles using weighted majority voting with local accuracy estimates».
21. X. Wang, «A New Model for Measuring the Accuracies of Majority Voting Ensembles,» de *IEEE World Congress on Computational Intelligence*, Brisbane, 2012.
22. L. Kovacs, H. Toman, A. Jonas, L. Hajdu y A. Hajdu, «Generalizing the majority voting scheme to conditional voting».
23. G. Brown y L. I. Kuncheva, «"Good" and "Bad" Diversity in Majority Vote Ensembles,» de *Multiple Classifier Systems*, Springer, Heidelberg, 2010, pp. 124-133.
24. Y.-S. Chung, D. F. Hsu y C. Y. Tang, «On the Diversity-Performance Relationship for Majority Voting in Classifier Ensembles,» de *Multiple Classifier Systems*, Springer, 2007, pp. 407-420.
25. P. Hong, L. Chengde, L. Linkai y Z. Qifeng, «Accuracy of Classifier Combining Based on Majority Voting,» de *IEEE International Conference on Control and Automation*, 2007.
26. A. Narasimhamurthy, «Theoretical Bounds of Majority Voting Performance for a Binary Classification Problem,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, nº 12, pp. 1988--1995, 2005.
27. L. I. Kuncheva, C. J. Whitaker y S. C. A., «Limits on the Majority Vote Accuracy in Classifier Fusion,» *Pattern Analysis and Applications*, vol. 6, nº 1, pp. 22-31, 2003.
28. D. Ruta y B. Gabrys, «A theoretical analysis of the limits of majority voting errors for multiple classifier systems,» *Pattern Analysis & Applications*, vol. 5, nº 4, pp. 333--350, 2002.
29. S. Site y S. K. Mishra, «Model for measuring accuracies of majority voting of Ensemble Classifier with COB and Genetic algorithm,» de *2013 International Conference on Information Communication and Embedded Systems (ICICES)*, IEEE, 2013, pp. 99-103.
30. L. Lam y C. Y. Suen, «Application of Majority Voting to Pattern Recognition: An Analysis of Its Behavior and Performance,» *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, vol. 27, nº 5, pp. 553-568, 1997.
31. N. C. de Condorcet, «Essai sur l'Application de l'Analyze `a la Probabilit´e des D´ecisions Rendues `a la Pluralit´e des Voix,» 1785.
32. Z. H. Zhou, Ensemble Methods Foundations and Algorithms, Chapman & Hall, 2012.
33. W. Pierce, «Improving reliability of digital systems by redundancy and adaptation,» PhD Thesis, Electrical Engineering, Stanford University, 1961.
34. D. M. Titterington, G. D. Murray, M. L. S., D. J. Spiegelhalter, A. M. Skene, J. D. F. Habbema y G. J. Gelpke, «Comparison of discriminant techniques applied to a complex data set of head injured patients,» *Journal of the Royal Statistical Society*, pp. 145-175, 1981.
35. Y. S. Huang y C. Y. Suen, «A method of combining multiple experts for the recognition of unconstrained handwritten numerals,» *IEEE Transactions on pattern analysis and machine intelligence*, vol. 17, nº 1, pp. 90-94, 1995.
36. Y. S. Huang y C. Y. Suen, «The behavior-knowledge space method for combination of multiple classifiers,» *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 347-347, 1993.
37. S. Raudys y F. Roli, «The Behavior Knowledge Space Fusion Method: Analysis of Generalization Error and Strategies for Performance Improvement,» de *Multiple Classifier Systems*, Springer, 2003, pp. 55-64.
38. F. Roli, S. Raudys y G. L. Marcialis, «An Experimental Comparison of Fixed and Trained Fusion Rules for Crisp Classifier Outputs,» de *Multiple Classifier Systems*, Springer, 2002, pp. 232-241.
39. K. D. Wernecke, «A coupling procedure for discrimination of mixed data.,» *Biometrics*, vol. 48, pp. 497--506, 1992.
40. T. K. Ho, J. J. Hull y S. N. Srihari, «Decision Combination in Multiple Classifier Systems,» *IEEE*

Transactions on pattern analysis and machine intelligence, vol. 16, nº 1, pp. 66-75, 1994.

41. D. Black, The Theory of Committees and Elections, London: Cambridge University Press, 1963.
42. L. Lam y C. Y. Suen, «Optimal combinations of pattern classifiers,» *Pattern Recognition Letters*, vol. 16, nº 9, pp. 945--954, 1995.
43. S. B. Cho, «Pattern recognition with neural networks combined by genetic algorithm,» *Fuzzy sets and systems*, vol. 103, nº 2, pp. 339-347, 1999.
44. S. Hashem, «Optimal Linear Combinations of Neural Networks,» *Neural Networks*, vol. 10, nº 4, pp. 599-614, 1997.
45. N. Ueda, «Optimal Linear Combination of Neural Networks for Improving Classification Performance,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, nº 2, pp. 207-215, 2000.
46. D. H. Sattinger, Measure Theory & Integration, Department of Mathematics, Yale University, 2004.
47. M. Sugeno, «Theory of fuzzy integrals and its applications,» Tesis Doctoral, 1974.
48. L. I. Kuncheva, «"Fuzzy" vs "Non-fuzzy" in combining classifiers: An Experimental Study».
49. Y. S. Huang y C. Y. Suen, «A method of combining multiple classifiers -a neural network approach,» de *12th International Conference on Pattern Recognition* , Jerusalem, Israel, 1994.
50. L. I. Kuncheva, J. C. Bezdek y R. P. Duin, «Decision Templates for Multiple Classifier Fusion: An Experimental Comparison,» *Pattern Recognition*, vol. 2, nº 34, pp. 299-314, 2001.
51. B. V. Dasarathy y B. V. Sheela, «A composite classifier system design: concepts and methodology,» *Proceedings of the IEEE*, vol. 67, nº 5, pp. 708-713, 1979.
52. K. Woods, K. W. P. y K. Bowyer, «Combination of Multiple Classifiers Using Local Accuracy Estimates,» de *Proceedings CVPR'96, 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1996.* , 1997.
53. M. García y J. Ruiz-Shulcloper, «Selecting Prototypes in Mixed Incomplete Data,» de *Progress in Pattern Recognition, Image Analysis and Applications*, Springer, 2005, p. 450–459.
54. G. Giacinto y F. Roli, «Design of effective neural network ensembles for image classification processes,» *Image and Vision Computing*, vol. 19, nº 9, pp. 699-707, 2001.
55. L. I. Kuncheva, «Clustering-and-selection model for classifiers combination,» de *Proc. Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, Brighton, UK, 2000.
56. L. I. Kuncheva, «Switching between selection and fusion in combining classifiers,» *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 32, nº 2, pp. 146-156, 2002.
57. L. Breiman, «Bagging predictors,» *Machine Learning*, vol. 24, nº 2, pp. 123-140, 1996.
58. B. Efron y R. Tibshirani, An Introduction to the Bootstrap, New York: Chapman & Hall, 1993.
59. P. Buhlmann y B. Yu, «Analyzing Bagging,» *The Annals of Statistics*, vol. 30, nº 4, pp. 927-961, 2002.
60. L. Breiman, «Arcing Classifiers,» Annals of Statistics, 1998.
61. L. Breiman, «Out-of-bag estimation,» Technical report, Department of Statistics, University of California, 1996.
62. P. Domingos, «Why does Bagging works? A bayesian account and its implications,» de *KDD*, Citeseer, 1997, pp. 155-158.
63. J. H. Friedman y P. Hall, «On Bagging and nonlinear estimators,» *Journal of statistical planning and inference*, vol. 137, nº 3, pp. 669-683, 2007.
64. Z. Cao, P. A. Papakonstantinou y J. Xu, «Bagging by design (on the suboptimality of bagging),» Tsinghua University, 2014.
65. T. K. Ho, «The random subspace method for constructing decision forests,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, nº 8, pp. 832-844, 1998.
66. R. Bryll, R. Gutierrez-Osuna y F. Quek, «Attribute bagging: improving accuracy of classifiers ensembles by using random features subsets,» *Pattern Recognition*, vol. 36, nº 6, pp. 1291--1302, 2003.
67. T. G. Dietterich, «A experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization,» *Machine learning*, vol. 40, nº 2, pp. 139-157, 2000.
68. L. Breiman, «Random forests,» *Machine Learning*, vol. 45, nº 1, pp. 5-32, 2001.

69. C. Xiong, D. Johnson, R. Xu y J. J. Corso, «Random forest for metric learning with implicit pairwise position dependence,» de *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012.
70. D. Zikic, B. Glocker y A. Criminisi, de *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2012.
71. M. Denil, D. Matheson y N. de Freitas, «Narrowing the Gap: Random forests In Theory and In Practice,» 2013.
72. L. Breiman, «Consistency for a simple model of random forest,» Tech. Rep. 670, Statistics Department, University of California, Berkeley, CA, USA, Berkeley, 2004.
73. G. Biau, L. Devroye y G. Lugosi, «Consistency of random forests and other averaging classifiers,» *Journal of Machine Learning Research*, vol. 9, pp. 2015-2033, 2008.
74. G. Biau, «Analysis of a Random Forests model,» *Journal of Machine Learning Research*, vol. 13, pp. 1063-1095 , 2012.
75. M. Denil, D. Matheson y N. de Freitas, «Consistency of online random forests,» de *International Conference on Machine Learning*, 2013.
76. R. E. Schapire, «The strength of weak learnability,» *Machine learning*, vol. 5, nº 2, pp. 197-227, 1990.
77. M. Kearns y L. G. Valiant, «Cryptographic limitations on learning boolean formulae and finite automata,» de *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, 1989.
78. Y. Freund y R. E. Schapire, «A decision-theoretic generalization of on-line learning and and an application to boosting,» *Journal of computer and system sciences*, vol. 55, nº 1, pp. 119-139, 1997.
79. J. Friedman, T. Hastie y R. Tibshirani, «Additive logistic regression: A statistical view of boosting (with discussions),» *The Annals of Statistics*, pp. 337-374, 2000.
80. R. E. Schapire, Y. Freund, P. Bartlett y W. S. Lee, «Boosting the margin: A new explanation for the effectiveness of voting methods,» *The annals of statistics*, vol. 26, nº 5, pp. 1651--1686, 1998.
81. J. R. Quinlan, «Bagging, Boosting, and C4.5,» de *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.
82. P. Viola y M. Jones, «Fast and robust classification using asymmetric adaboost and a detector cascade,» *Advances in Neural Information Processing Systems*, vol. 2, pp. 1311-1318, 2002.
83. E. Bauer y R. Kohavi, «An empirical comparison of voting classification algorithms: Bagging, boosting, and variants,» *Machine Learning*, vol. 36, nº 1-2, pp. 105-139, 1999.
84. A. Criminisi, J. Shotton y J. Konukoglu, «Decision forests: A unified framework for classision, density estimation, manifold learning and semisupervised learning,» *Foundations and Trends in Computer Graphics and Vision*, vol. 7, nº 2-3, pp. 81-227, 2011.
85. L. K. Hansen y P. Salomon, «Neural Network Ensembles,» *IEEE Transactions on pattern analysis and machine intelligence*, vol. 12, nº 10, pp. 993-1001, 1990.
86. W. Wang, P. Jones y D. Partridge, «Diversity between neural networks and decision trees for building multiple classifiers systems,» de *International Workshop on Multiple Classifiers Systems*, Calgiari, Italy, 2000.
87. R. E. Schapire y Y. Singer, «Improved boosting algorithms using confidence-rated predictions,» *Machine learning*, vol. 37, nº 3, pp. 297--336, 1999.
88. T. Hastie y R. Tibshirani, «Classification by pairwise coupling,» *The Annals of Statistics*, vol. 26, nº 2, pp. 451-471, 1998.
89. T. G. Dietterich y G. Bakiri, «Error-correcting output codes: A general method for improving multiclass inductive learning programs,» de *9th National Conference on Artificial Intelligence*, 1991.
90. L. Didaci, G. Fumera y F. Roli, «Diversity in Classifier Ensembles: Fertile Concept or Dead End?,» de *Multiple Classifier Systems*, Springer, 2013, pp. 37-48.
91. C. A. Shipp y L. Kuncheva, «Relationships between combination methods and measures of diversity in combining classifiers,» *Information fusion*, vol. 3, nº 2, pp. 135-148, 2002.
92. A. Sharkey y N. Sharkey, «Combining diverse neural networks,» *The Knowledge Engineering Review*, vol. 12, nº 3, pp. 231-247, 1997.

93. G. Brown, J. Wyatt, R. Harris y X. Yao, «Diversity creation methods: a survey and categorisation,» *Information Fusion,* vol. 6, nº 1, pp. 5-20, 2005.
94. D. B. Skalak, «The sources of increased accuracy for two proposed boosting algorithms,» de *American Association for Artificial Intelligence*, 1996.
95. G. Yule, «On the association of attributes in statistics,» *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character,* vol. 194, nº 252-261, pp. 257--319, 1900.
96. P. H. A. Sneath y R. R. Sokal, Numerical Taxonomy: The Principles and Practice of Numerical Classification, San Francisco: W. H. Freeman, 1973.
97. L. Kuncheva y C. Whitaker, «Measures of diversity in classifier ensembles and their relationship with ensemble accuracy,» *Machine learning,* vol. 51, nº 2, pp. 181-207, 2003.
98. E. K. Tang, P. N. Suganthan y Y. X., «An analysis of diversity measures,» *Machine Learning,* vol. 65, nº 1, pp. 247-271, 2006.
99. N. Ueda y R. Nakano, «Generalization error of ensemble estimators,» de *International Conference on Neural Networks*, 1996.
100. A. Krogh y J. Vedelsby, «Neural network ensembles, cross validation, and active learning,» de *NIPS 7*, 1995.
101. Y. Liu, «Negative correlation learning and evolutionary neural network ensembles,» Ph.D. thesis, University College, The University of New South Wales, Canberra, 1998.
102. G. Brown y J. L. Wyatt, «The use of the ambiguity decomposition in neural network ensemble learning methods,» de *20th International Conference on Machine Learning*, Washington, 2003.
103. A. Sharkey, «Types of multinet system,» de *Workshop on Multiple Classifier Systems (LNCS 2364)*, Calgiari, 2002.
104. T. K. Ho, «Data complexity analysis for classifier combination,» de *Workshop on Multiple Classifier Systems (LNCS 2096)*, Cambridge, 2001.
105. T. K. Ho, «Multiple classifier combination: Lessons and the next steps,» de *Hybrid Methods in Pattern Recognition*, World Scientific Publishing, 2002, pp. 171-198.
106. G. Valentini y F. Masulli, «Ensembles of learning machines,» *Neural Nets.*, pp. 3-20, 2002.
107. R. P. Duin, «The combining classifiers: to train or not to train?,» de *International Conference on Pattern Recognition*, Canada, 2002.
108. M. S. Kamel y N. M. Wanas, «Data dependence in combining classifiers,» de *Workshop on Multiple Classifier Systems (LNCS 2709)*, Guildford, 2003.

RT_064, octubre 2014
Aprobado por el Consejo Científico CENATAV

Editor: Lic. Lucía González Bayona
Diseño de Portada: Di. Alejandro Pérez Abraham
RNPS No. 2142
ISSN 2072-6287
Indicaciones para los Autores:
Seguir la plantilla que aparece en www.cenatav.co.cu
C E N A T A V
7ma. A No. 21406 e/214 y 216, Rpto. Siboney, Playa;
La Habana. Cuba. C.P. 12200
Impreso en Cuba