Un nuevo clasificador de préstamos bancarios a través de la minería de datos

M. Beltrán Pascual, Á. Muñoz Alamillos, A. Muñoz Martínez

Abstract

En este artículo se muestra la forma de implementar eficientemente un nuevo clasificador de préstamos bancarios a través de la minería de datos. Los modelos de credit scoring ayudan en un primer momento a la toma de decisión a los gerentes de los bancos.

La metodología propuesta es la combinación de modelos. Los multiclasificadores son una excelente forma de integrar la información de diferentes fuentes. Esta combinación de dos o más clasificadores, en general, proporciona estimaciones más robustas y eficientes que cuando se utiliza un único clasificador. También se utilizan porque resuelven el problema de sobreadaptación (overfitting) y es posible obtener buenos resultados con pocos datos.

Se describen los principales métodos de combinación de clasificadores utilizados en minería de datos: Bagging, Boosting, Stacking, Cascading… y se aplican a los datos de clientes bancarios obteniéndose unos excelentes resultados comparados con los modelos y algoritmos utilizados individualmente.

Palabras claves: minería de datos, multiclasificadores, credit scoring, curva ROC

1. Introducción.

La minería de datos tiene como finalidad el descubrimiento de estructuras subyacentes escondidas en las bases de datos. Constituye un proceso iterativo que combina la experiencia sobre un problema dado con variedad de técnicas estadísticas tradicionales de análisis de datos y de tecnología avanzada de aprendizaje automático.

En este artículo se muestra la forma de implementar un nuevo clasificador de préstamos bancarios a través de la minería de datos. Este es un problema que se concentra en predecir a qué clase pertenecen las muestras evaluadas. Con los datos que aporta el cliente que solicita el crédito, la información de la base de datos y la que el modelo sugiere, el gerente del banco puede optimizar su decisión de conceder o no el crédito solicitado. Sin olvidar que los modelos de credit scoring ayudan en un primer momento a la toma de decisión de si conceder o no el crédito, a la vez que lo pueden justificar, pero hay que tener en cuenta que existe otra dimensión cualitativa en la toma de la decisión que no se puede abordar en los modelos matemáticos.

Disponer de un buen método que nos ayude a tomar decisiones más correctas puede ayudar a ahorrarse mucho dinero a la organización y, mucho más, en la actual situación donde a las entidades del sector financiero se les está exigiendo mayor disponibilidad de dinero.

Para la realización de este trabajo se ha dispuesto de una parte de la base de datos de los clientes de una entidad bancaria real que han solicitado un crédito.

Las formas de abordar el problema de la clasificación son variadas. La gran variedad de técnicas existentes pueden incorporar análisis estadísticos, herramientas de minería de datos, inteligencia artificial o aprendizaje de máquina. Una de ellas, la más clásica, en los problemas de credit scoring ha sido a través de la regresión logística que estadísticamente ofrece buenos resultados. Otro enfoque clásico es sintetizar la información de la base de datos de clientes a través de reglas y de árboles de decisión. Otras aproximaciones para resolver el problema de credit scoring se basan en redes neuronales, aplicando los algoritmos evolutivos, splines de regresión adaptativa, las maquinas de vectores soporte o de la lógica borrosa, además de otras técnicas novedosas.

La metodología utilizada en este artículo, para obtener un mayor porcentaje de aciertos en este problema de clasificación, es a través de los multiclasificadores, que en la literatura existente se conoce con muchos nombres: métodos de ensamble, modelos múltiples, sistemas de múltiples clasificadores, combinación de clasificadores, integración de clasificadores, mezcla de expertos, comité de decisión, fusión de clasificadores y aprendizaje multimodelo.

La combinación de dos o más clasificadores, en general, proporciona estimaciones más robustas y eficientes que cuando se utiliza un único clasificador. También se emplean porque resuelven el problema de sobreadaptación (overfitting) y, además, es posible obtener buenos resultados cuando la base de datos no es muy grande.

En las páginas que siguen se describen de manera somera, algunos de los principales algoritmos utilizados en problemas de clasificación de manera individual y, posteriormente, se detallan, los multiclasificadores más utilizados en minería de datos. Estos algoritmos descritos, así como otros más, se aplican a una base de datos de clientes bancarios.

1.1 Descripción de la base de datos.

El conjunto de datos estudiado contiene 1.767 registros que representan a los clientes de una caja de Ahorros de la Rioja que demandaron un crédito. Existen atributos numéricos y nominales. Los atributos de cada cliente nos informan sobre diversas cuestiones: estado civil, sexo, edad, tipo de trabajo, código de profesión, situación de la vivienda nacionalidad, etcétera., así como otra información relacionada con el crédito: finalidad, importe solicitado, importes pendientes en su entidad bancaria y en otras, patrimonio, valor neto de la vivienda, situación de ingresos, cuotas y gastos de alquiler y préstamos, etcétera. También sabemos si el crédito se ha concedido o se ha denegado.

2. Preprocesado de los datos.

Es bastante obvio que para obtener buenos resultados en los análisis estadísticos y de minería de datos se debe de partir de una base de datos con información consistente, completa, comprensible y limpia para que los análisis sean útiles. Es necesario por tanto que los datos sean analizados con conciencia-

La tarea del preprocesado de los datos tiene como objetivo obtener una vista minable, es decir con aquel conjunto de datos lo suficientemente libre de errores, ya filtrados, sin datos anómalos y cuyas variables sean lo más independientes entre sí.

Las tareas que conlleva este paso del procedimiento de la minería de datos, según diversos autores, abarca un considerable tiempo, entre el 70% y el 90 % del tiempo total destinado a un proyecto de minería de datos.

Las tareas de limpieza y transformación de los datos hasta conseguir un Base de datos minable son numerosas pero las principales se pueden resumir en las siguientes: Imputación de datos ausentes, filtrado y eliminación de valores anómalos o outliers, transformación de la base de datos, reducción de variables o de la dimensionalidad y balanceo de la base de datos.

2.2 Reducción de variables o de la dimensionalidad.

El alto número de variables recogidas para el estudio de un fenómeno a veces es un problema para el aprendizaje si el número de instancias o ejemplos de la muestra es reducido. Este es el problema conocido como la maldición de la multidimensionalidad.

La selección de atributos es uno de los problemas más complejos al que pretende hacer frente el aprendizaje automático. El objetivo es eliminar variables redundantes, atributos espúreos y en general todos aquellas variables donde su presencia en la base de datos no aporte un aumento de la información de la misma. En bases de datos con muchas características y pocas instancias resulta imposible construir modelos. Cuantas menos sean las variables, más fácil será de entender. Por otra parte, con menos variables mejoramos la visualización de los datos.

En la literatura de selección de variables existen dos métodos generales para escoger las mejores características de la base de datos: métodos de filtro y métodos basados en modelos. En los primeros se filtran los atributos irrelevantes antes de aplicar las técnicas de minería de datos. El criterio que establece las variables óptimas se basa en una medida de calidad que se calcula a partir de los datos mismos. En los métodos basados en modelos, también conocidos como métodos de envolvente o wrapper, la bondad de la selección de las variables se evalúa a través de un modelo utilizando, lógicamente, un método de validación.

En el caso de la selección de atributos debemos definir un algoritmo que evaluará cada atributo individualmente del conjunto de datos inicial, que se denomina “attribute evaluator” y un método de búsqueda que hará una búsqueda en el espacio de posibles combinaciones de todos los subconjuntos del conjunto de atributos.

De esta forma podremos evaluar independientemente cada una de las combinaciones de atributos y, con ello, seleccionar aquellas configuraciones de atributos que maximicen la función de evaluación de atributos.

Para resolver el problema de plantear combinaciones de atributos la función que evalúa cada subconjunto de atributo es preciso utilizar un algoritmo de búsqueda que recorra el espacio de posibles combinaciones de una forma organizada, o adecuada al problema.

Además de las componentes principales existen dos tipos de evaluadores: evaluadores de subconjuntos o selectores (SubSetVal) y prorrateadores de atributos (AttributeEval).

Los SubSetVal necesitan una estrategia de búsqueda (Search Method) y los AttributeEval ordenan las variables según su relevancia, así que necesitan un Ranker.

Habitualmente en las situaciones en la que se emplea selección de atributos no es posible hacer un recorrido exhaustivo en el espacio de combinaciones por lo que la selección adecuada de un algoritmo de búsqueda resulta crítica.

Para esta base de datos se utiliza, en primer lugar, el algoritmo evaluador de atributos "CfsSubsetEval" que dispone el programa WEKA. Este algoritmo es el más sencillo, ya que puntúa a cada atributo en función de su entropía. Como algoritmo de búsqueda utilizamos los algoritmos genéticos. En segundo lugar recurrimos al método Ranker para que nos facilite una ordenación de los atributos según su importancia.

Los algoritmos genéticos propuestos por [Holland.1975], suponen uno de los enfoques más originales en la minería de datos, se inspiran en el comportamiento natural de la evolución, para ello se codifica cada uno de los casos de prueba como una cadena binaria (que se asemejaría a un gen). Esta cadena se replica o se inhibe en función de su importancia, determinada por una función denominada de ajuste o fitness.

Los algoritmos genéticos son adecuados para obtener buenas aproximaciones en problemas de búsqueda, aprendizaje y optimización [Marczyk. (2004)].

De forma esquemática un algoritmo genético es una función matemática que tomando como entrada unos individuos iniciales (población origen) selecciona aquellos ejemplares (también llamados genes) que recombinándose por algún método generarán como resultado la siguiente generación. Esta función se aplicará de forma iterativa hasta verificar alguna condición de parada, bien pueda ser un número máximo de iteraciones o bien la obtención de un individuo que cumpla unas restricciones iniciales.

Con la combinación de diferentes estrategias: ranker y algoritmos genéticos se eliminan cuatro variables que no son significativas para el análisis de los resultados. En los cuadros de resultados que se presentan los resultados obtenidos se presentan solamente para el conjunto de las 15 variables que se han considerado significativas para la clasificación.

2.3 Balanceo de las clases.

A la hora de aplicar los métodos de clasificación hemos de tener en cuenta cómo están distribuidas las instancias respecto a la clase. Al no estar balanceadas las clases los clasificadores estarán sesgados a predecir un porcentaje más elevado de la clase más favorecida.

Para observar el efecto que se produce en el porcentaje de acierto, según la clase, en los dos cuadros siguientes se presentan, para diversos métodos de clasificación, el porcentaje correctamente clasificado tanto para el total como para cada una de las clases.

CUADRO Nº 1. Muestra desbalanceada (1.565 instancia clase SI y 167 clase NO)

Técnica	CLASE SI (%)	CLASE NO (%)	TOTAL (%)	Estadístico KAPPA	AREA ROC
C 4.5	96,0	35,9	90,2	0,363	0,733
Maq. Vect. Soporte	100,0	0,0	90,4	0,000	0,500
Perceptrón Mult.	96,0	29,9	89,6	0,303	0,806
Redes Base Radial	98,7	10,2	90,2	0,137	0,831
NAÏVE BAYES	54,8	86,2	57,8	0,145	0,811
Red Bayesiana(TAN)	94,3	50,3	90,1	0,439	0,890
Red Bayesiana(K2)	93,2	51,5	89,2	0,419	0,894
Regresión Logística	97,7	33,5	91,5	0,391	0,883
Metaclasificadores					
Random Forest	98,8	26,3	91,8	0,348	0,866
ADABOOST	95,1	38,1	90,2	0,424	0,808
BAGGING	95,1	49,1	90,7	0,453	0,900
STAKING C (5 modelos)	97,8	36,5	91,9	0,423	0,889
Random Committee	98,5	32,9	92,2	0,413	0,869
RandomSubSpace	99,3	17,4	91,4	0,252	0,865
Incorporación de costes					
Metacost (9,6/1))	80,3	78,4	80,1	0,340	0,877

El tamaño de la muestra juega un papel determinante en la bondad de los modelos de clasificación. Cuando el desbalanceo es considerable descubrir regularidades inherentes a la clase minoritaria se convierte en una tarea ardua y de poca fiabilidad. Japkowicz and Stephen (2009) concluyen que si los dominios son separables linealmente no son sensibles al problema del desequilibrio de las clases.

En el ejemplo que estamos tratando podemos ver que cuando mantenemos la base de datos con las clases desequilibradas todos los métodos presentan una importante diferencia de aciertos entre las clases.

Los métodos de clasificación favorecen en general a la clase mayoritaria salvo en el caso del clasificador bayesiano Naïves Bayes que clasifica mejor a la clase minoritaria. Se da el caso extremo en el que un clasificador, las máquinas de vectores soporte, clasifican correctamente a todos de la clase mayoritaria y a ninguno de la minoritaria. Tampoco los metaclasificadores estiman correctamente ambas clases. Solamente introduciendo un método cuyo aprendizaje sea sensible al coste se logra equilibrar la precisión de los ejemplo bien clasificados.

Las soluciones para tratar el desbalanceo se pueden encuadrar en dos grupos: soluciones a nivel de datos y a nivel de algoritmos.

Las técnicas dirigidas a modificar los datos tratan de remuestrear las tallas de entrenamiento, bien sea a través de sobremuestreo de la clase minoritaria o el submuestreo de la clase que tiene mayores instancias. Aunque estas técnicas han demostrados su efectividad no dejan de tener ciertos inconvenientes: pueden eliminar ejemplos útiles e incrementar los costes. Otra crítica a esta estrategia se refiere al cambio que se realiza en la distribución original del conjunto de entrenamiento de los datos

En el cuadro nº 2 se expresan los resultados de diferentes clasificadores aplicados a una muestra donde se han balanceado ambas clases. Cuando existe equilibrio de las instancias en la base de datos los porcentajes de acierto de los clasificadores para ambas clases están mucho más igualados.

El tema de muestras desbalanceadas se ha tratado extensamente y se han utilizado muchas estrategias, aunque se puede afirmar que no existe una solución concluyente sobre qué solución es mejor. Hulse y otros (2007) concluyen que la decisión sobre la mejor técnica está influenciada en gran medida por la naturaleza del clasificador y la medida de efectividad.

Otra forma que disponemos para para combatir el desbalance de clases, es a través del establecimientos de una matriz de costes, lo que se ha llamado método del costo-sensitivo (*cost-sensitive*). Este método se basa en la aseveración de que el precio de cometer un error de clasificación debe ser distinto para cada clase. Es evidente que en este ejemplo no es lo mismo conceder un crédito y no pagarlo que no concederlo cuando se debería haber concedido.

CUADRO Nº 2. Muestra equilibrada (167 ejemplos para cada clase)

Técnica	%CLASE SI	%CLASE NO	% CLASE TOTAL	Estadístico KAPPA	ROC AREA
C 4.5	78,4	80,8	79,6	0,593	0,807
Maq. Vect. Soporte	80,2	76,0	78,1	0,563	0,781
Perceptrón Mult.	78,4	80,2	79,3	0,587	0,847
Redes Base Radial					
NAÏVE BAYES	55,7	86,2	71,0	0,419	0,793
Red Bayesiana(TAN)	78,4	82,0	80,2	0,605	0,867
Red Bayesiana(K2)	79,0	79,0	79,0	0,581	0,865
Regresión Logística	79,0	78,4	78,7	0,575	0,867
Metaclasificadores					
Random Forest	78,4	82,0	80,2	0,605	0,865
ADABOOST	80,2	83,2	81,7	0,635	0,871
BAGGING	80,8	78,4	79,6	0,593	0,879
STAKING C (5 modelos)	79,6	82,0	80,8	0,617	0,893
Random Committee	81,4	79,6	80,5	0,611	0,893
RandomSubSpace	77,2	82,6	79,9	0,5988	0,861

La técnica más sencilla de sobremuestreo es la aleatoria a través de la réplica de ejemplos en la misma clase, pero este método puede ocasionar un alto sobreajuste de los clasificadores.

Como técnica más inteligente para incrementar los ejemplos de la clase minoritaria se encuentra el algoritmo SMOTE (Synthetic Minoritary Over-sampling TEchnique) originario de Chawla y otros (2002). En este método la creación de nuevas muestras se origina a través de la interpolación. En un primer paso elegimos los K vecinos más cercanos y que pertenecen a su misma clase. Posteriormente elegimos el número de muestras artificiales que se generarán y, finalmente, para generar una nueva muestra, se calcula la diferencia entre el vector de atributos bajo consideración y uno de los vecinos más cercanos de los k vecinos elegidos al azar. El resultado de la diferencia se multiplica por un valor aleatorio entre cero y uno.

El algoritmo SMOTE se ha modificado de diferentes maneras para adaptarse mejor a muchos ejemplos. Algunas de estas aportaciones son las efectuadas por Han y otros (2005) que proponen el algoritmos Borderline-SMOTE para generar ejemplos positivos cercanos a dicha frontera. Wang y otros (2006) presentan el algoritmo LLE-SMOTE (Locally Linear Embedding) que proyecta conjuntos de alta dimensionalidad a otro de menor dimensionalidad. En este espacio de reducida dimensionalidad es donde se aplica SMOTE y después los ejemplos generados son transformados a su espacio de representación original.

Otras formas de obtener una representación mayor de la clase minoritaria se basan en técnicas de agrupamiento, por ejemplo Japkowicz (2001) emplea el algoritmo de clustering k-medias sobre cada clase por separado. Los clusters resultantes se sobremuestrean aleatoriamente hasta conseguir un equilibrio entre las clases. Otro trabajo en esta línea de investigación es el de Cohen y otros (2006) que también explora la generación de nuevas instancias a través de algoritmos de clustering, pero los centroides de los clusters se obtienen a través de un algoritmo aglomerativo jerárquico.

En cuanto a las técnicas de submuestreo una de las primeras propuestas para editar o filtrar las muestras de entrenamiento fue el algoritmo de Edición de Wilson (1972), también conocido como la regla del vecino más cercano editado (Edited Nearest Neighbor). Actualmente existen muchas formas de proceder, algunas de ellas son las siguientes: a través del submuestreo aleatorio de Ho y Japkowicz, (2004), con submuestreo dirigido, algoritmo One-sides selection de Kubat y Matwin, (1997), con técnicas de vecindad, algoritmo Neighbordhood Cleaning Rule de Laurikkala, (2002). Con submuestreo aplicando algoritmos genéticos, Kuncheva y Jain, (1999), con submuestreo por distancia, Zhanng y Mani, (2003), con submuestreo por clustering, Cohen y otros, (2006), a través del aprendizaje activo de Provost, (2003). Respecto a los métodos de clasificación en entornos no balanceados que no cambian la

distribución a priori de las clases nos encontramos con las soluciones a nivel de algoritmos: Aprendizaje sensible al coste, algoritmos de clasificación con sesgo hacia la clase minoritaria y los clasificadores de una clase.

En este trabajo el clasificador que se aplica para poder comparar con el resto de los algoritmos es el metacost [Domingos, (1999)] que se encuentra incluido en Weka. El objetivo de este procedimientos es reetiquetar cada muestra de entrenamiento por la estimación del riesgo de Bayes. Finalmente el clasificador se entrena con un método no basado en costes con el conjunto que ya ha sido reetiquetado.

2.4 Métodos de coste sensitivo para clasificación.

En el problema de la concesión de créditos a través del credit scoring los errores de predicción causarían costos desiguales dado que, supuestamente, desde el punto de vista del banco, es mucho más costoso que el clasificador se equivoque ofreciendo un crédito a una persona que no lo devuelve, que la situación contraria, denegar un crédito a un cliente que sí lo devolvería.

CUADRO Nº 3. Fichero con SMOTE y CUBO. Incorporación de los costes

Fase de entrenamiento	CLASE SI (%)	CLASE NO (%)	TOTAL (%)	Estadístico KAPPA	AREA ROC
Metacost (1/1))	85,3	84,5	84,9	0,698	0,923
Metacost (2/1))	81,1	86,8	83,9	0,679	0,922
Metacost (3/1))	79,5	89,7	84,6	0,691	0,926
MetacostSensitive (1/1)	83,7	83,2	83,4	0,669	0,914
MetacostSensitive (2/1)	81,7	85,5	83,6	0,672	0,910
MetacostSensitive (3/1)	79,8	86,8	83,3	0,666	0,906
Fase de test					
Metacost (1/1))	78,6	83,3	80,8	0,615	0,899
Metacost (2/1))	78,6	91,7	84,6	0,694	0,869
Metacost (3/1))	78,6	91,7	84,6	0,694	0,875
MetacostSensitive (1/1)	85,7	100,0	92,3	0,847	0,940
MetacostSensitive (2/1)	78,6	91,7	84,6	0,694	0,905
MetacostSensitive (3/1)	71,4	91,7	80,8	0,620	0,905

Para este tipo de problemas, la información sobre los costes de los errores viene expresada a través de una matriz de costes. En este tipo de matrices se recoge el coste de cada una de las posibles combinaciones entre la clase predicha por el modelo y la clase real

Una de las maneras de resolverlo es, una vez establecida la matriz de costes, evaluar cada clasificador multiplicando los resultados de la matriz de confusión por cada uno de los costes asociados. Desde esta perspectiva el mejor clasificador sería aquel que arrojara el menor coste.

Otro enfoque distinto es utilizar un método de aprendizaje que esté basado en la reducción del coste en vez de incrementar la precisión. Weka contempla dos algoritmos cuyos resultados tanto en la fase de entrenamiento como en la de test se reflejan en el cuadro nº 3. Se puede observar que obtienen un grado elevado de aciertos. Cuando en la matriz de coste se establece un coste triple para la clase NO respecto de la clase SI, Falsos positivos frente a falsos negativos, el procedimiento MetacostSensitive, acierta el 100% la clase NO y, también, es muy preciso pronosticando la otra clase. Se observan valores altos tanto en el estadístico Kappa como en la curva ROC.

En esta investigación los resultados de los diferentes clasificadores que se presentan se aplican a un conjunto de datos que se han balanceado a través de un método mixto donde se aplica el método SMOTE a la clase minoritaria y se reduce la muestra de la clase mayoritaria a través del método del submuestreo equilibrado del cubo, propuesto por Deville y Tillé (2004). Este método de muestreo es el único que nos permite seleccionar una muestra equilibrada sobre variables auxiliares con probabilidades de inclusión iguales o no. El método del cubo selecciona únicamente las muestras cuyos estimadores de Horvitz-Thompson son iguales a los totales de las variables auxiliares conocidas.

En nuestra base de datos las variables auxiliares son el estado civil, el tipo de trabajo y la nacionalidad

2.5 Evaluación de modelos.

En cualquier etapa de un proceso de minería de datos resulta fundamental el poder estimar los niveles de calidad obtenidos por el modelo que estemos construyendo. En función del problema y el modelo existen infinidad de mecanismos de evaluación.

Disponemos de un conjunto de individuos del que conocemos previamente la clase a la que pertenecen. Estos individuos los evaluamos en nuestro modelo y en función de la disparidad entre los resultados obtenidos y los datos reales podemos definir las siguientes métricas:

- **Porcentaje de acierto**: Número de casos acertados / número de casos totales. Esta es la métrica más sencilla y habitual pero resulta muy engañosa en la mayor parte de las veces pues no informa nada acerca de la distribución del error entre clases. Además siempre que apliquemos esta métrica debemos tener en cuenta el balanceo entre clases, como ya se ha comentado anteriormente. Si disponemos de una población en la que el 90% de los individuos son de clase A y el 10% restante de clase B, cualquier algoritmo que obtenga un porcentaje de acierto inferior al 90% (decir siempre que los

individuos son de clase A) diremos que no aporta ningún tipo de conocimiento en la clasificación.

- **Matriz de confusión**: La matriz de confusión supone un salto cualitativo respecto al porcentaje de acierto ya que nos permite conocer la distribución del error a lo largo de las clases. Una matriz de confusión estándar tiene la siguiente estructura:

		Clase predicha	
		si	no
Clase real	si	Verdaderos positivos	Falsos negativos
	no	Falsos positivos	Verdaderos negativos

```
=== Confusion Matrix ===

 a b   <-- classified as
 8 1 | a = yes
 1 4 | b = no
```

Cada una de las posibles 4 combinaciones de la tabla tienen un nombre propio. En el ejemplo del gráfico, de los 9 individuos que son de clase a, 8 han sido clasificados correctamente (verdaderos positivos) y 1 incorrectamente (falso negativo). De los 5 de la clase B, ha habido 1 falso positivo y 4 verdaderos negativos.

- **Estadístico Kappa**. Es un coeficiente estadístico que determina la precisión del modelo a la hora de predecir la clase verdadera.

Se define como:

$$k = \frac{P(A) - P(E)}{1 - P(E)}$$

Dónde P(A) es el porcentaje de acierto y P(E) es el porcentaje de casos cambiados.

- **Precisión**: Es el cociente de los ejemplos clasificados correctamente (verdaderos positivos) entre todos los elementos clasificados de esa clase (verdaderos positivos + falsos negativos)

- **Curvas ROC**: Las curvas ROC (Receiver Operating Characteristic) son gráficos que muestran la distribución de las fracciones de verdaderos positivos y la fracción de falsos negativos. Es el patrón de oro en muchas áreas de análisis de modelos ya que representan de forma compacta muchísima información del rendimiento de un clasificador.

Como forma agregada se suele utilizar el valor del área bajo la curva ROC, que es un valor entre 0,5 y 1 que nos permite evaluar el rendimiento de un modelo de manera muy precisa. El problema de las curvas ROC es que en ocasiones no podemos calcularlas para determinados algoritmos.

2.6 Validación cruzada.

De manera ideal siempre que desarrollemos un método clasificador lo óptimo es que lo entrenemos con un conjunto de entrenamiento y un conjunto de evaluación independientes que sean capaces de modelar la población original.

Como esta situación en muchas ocasiones no es factible por falta de datos, dificultad de obtener un muestreo adecuado, etcétera, se suele aplicar un esquema de validación denominado validación cruzada que consiste en dividir el conjunto de entrenamiento en k particiones, repetir el procedimiento de entrenamiento y validación k veces, de forma que en cada una de ellas se entrene el modelo con k-1 particiones y se evalúe con la partición restante.

Los resultados finales se suelen obtener por agregación de los resultados originales.

3. Principales métodos y algoritmos empleados en la clasificación.

3.1 Árboles de decisión

Los árboles de decisión son particiones secuenciales de un conjunto de datos que maximizan las diferencias de la variable dependiente. Nos ofrecen una forma concisa de definir grupos que son consistentes en sus atributos pero que varían en términos de la variable dependiente. Esta herramienta puede emplearse tanto para la resolución de problemas de clasificación cono de regresión: árboles de clasificación y árboles de regresión.

Mediante esta técnica se representan de forma gráfica un conjunto de reglas sobre las decisiones que se deben de tener en cuenta para asignar un determinado elemento a una clase (valor de salida).

A continuación se realiza una breve descripción de los principales algoritmos más utilizados y que podemos encontrar en la mayoría de los programas informáticos: AID, CART, CHAID, QUEST y el C5.0.

3.1.1 El algoritmo AID (Automatic Interaction Detection) o Detección Automática de Interaciones fue uno de los más utilizados en la década de los años setenta y principios de los ochenta hasta que surgió el CHAID. Estos dos algoritmos presentan dos limitaciones muy importantes, derivadas, por una parte, del elevado número de elementos muestrales que requieren para efectuar los análisis y, por otra, de la carencia de un modelo explícito que explique o determine la relación existente entre la variable dependiente y las variables explicativas.

3.1.2 El algoritmo CART, acrónimo de Classification And Regression Trees (Árboles de decisión y de regresión) fue diseñado por Breiman, Friedman, Losen y Stone (1984). Con este algoritmo se generan árboles de decisión binarios lo que quiere decir que cada nodo se divide en exactamente dos ramas. Este modelo admite variables de entrada y de salida nominales, ordinales y continuas por lo que se pueden resolver problemas de clasificación y de regresión.

3.1.3 El algoritmo QUEST, es el acrónimo de Quick, Unbiased, Efficient Statistical Tree (árbol estadístico eficiente, insesgado y rápido. Este método sólo puede ser utilizado si la variable de salida es categórica nominal.

3.1.4 El algoritmo CHAID, corrige muchas de las limitaciones del AID. Es un acrónimo de Chi-squared Automatic Interaction Detection (detector automático de interacciones mediante Ji cuadrado). Este algoritmo data desde 1980 y fue desarrollado por Kass. Aunque fue diseñado para trabajar sólo con variables categóricas, posteriormente se incluyó la posibilidad de trabajar con variables categóricas, nominales, categóricas ordinales y variables continuas, permitiendo generar tanto árboles de decisión para resolver problemas de clasificación como árboles de regresión. En este algoritmo los nodos se pueden dividir en más de dos ramas. La construcción del árbol se basa en el cálculo de la significación de un contrate estadístico como criterio para definir la jerarquía de las variables predictoras o de salida, al igual que para establecer las agrupaciones de valores similares respecto a las variables de salida a la vez que conserva inalterables todos los valores distintos. Todos los valores estadísticamente homogéneos son clasificados en una misma categoría y asignados a una única rama. Como medida estadística , si la prueba es continua, se utiliza la prueba F, mientras que si la variable predicha es categórica se utiliza la prueba Ji-cuadrado.

3.1.5 El algoritmo C5,0 crea modelos de árbol de clasificación, permitiendo sólo variables de salida categórica. Las variables de entrada pueden ser de naturaleza continua o categórica. La forma de inferir árboles de decisión a través de este algoritmo es el resultado de la evolución del algoritmo C4.5 (Quinlan, 1993) diseñado por el mismo autor y que a su vez es el núcleo del programa pertenece a la versión ID3 (Quinlan, 1986).

El algoritmo básico ID3 construye el árbol de decisión de manera descendente y empieza preguntándose, ¿qué atributo es el que debería ser colocado en la raíz del árbol?. Para resolver esta cuestión cada atributo es evaluado a través de un test estadístico que determina cómo clasifica él sólo los ejemplos de entrenamiento. Cuando se selecciona el mejor atributo éste es colocado en la raíz del árbol. Entonces una rama y su nodo se crea para cada valor posible del atributo en cuestión. Los ejemplos de entrenamiento son repartidos en los nodos descendentes de acuerdo al valor que tengan para el atributo de la raíz. El proceso se repite con los ejemplos para seleccionar un atributo que será ahora colocado en cada uno de los nodos generados.

Una de las maneras de cuantificar la bondad de un atributo consiste en considerar la cantidad de información que proveerá ese atributo tal y como está definido en la teoría de la información. Por tanto, este algoritmo está basado en el concepto de "ganancia de información". Para definir esta concepto necesitamos definir el concepto de entropía. La entropía es una medida de la incertidumbre que hay en un sistema, es decir, trata de medir ante una situación determinada la probabilidad de que ocurra cada uno de los posibles resultados. La función de entropía más utilizada es la binaria:

$$H_2(p,1-p) = (p * \log_2(p) + (1-p)\log_2(1-p)$$

Si el conjunto de los registros T se agrupan en función de las categorías de la variable de salida S, obteniéndose una proporción pk para cada grupo asociado a un posible resultado o categoría, la entropía la podemos expresar de la siguiente forma:

$$INFO(T) = -(p1 * \log_2(p1) + p2 * \log_2(p2) + ... + pk \log_2(pk)$$

Ahora se puede expresar la ganancia de información teniendo en cuenta una variable de entrada X:

$$GANANCIA(X,T) = INFO(T) - INFO(X,T) \qquad \text{donde}$$

$$INFO(X,T) = \sum_{i=1}^{k} \frac{T_i}{T} * INFO(T_i)$$

INFO(X,T) nos proporciona la información aportada por la variable de salida S cuando se tiene en cuenta una variable de entrada X.

INFO (X,T_i) es la entropía de la variable de salida S en cada subconjunto T_i determinado por la k categorías de la variable de entrada X. T_i es el número de registros asociados a una categoría i de la variable X.

3.2 Redes neuronales.

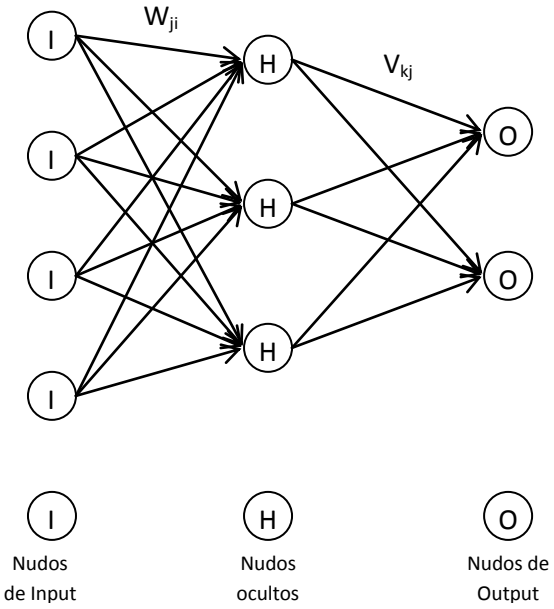
Las redes neuronales tratan de emular el comportamiento cerebral. Una red neuronal puede describirse mediante cuatro conceptos: el tipo de modelo de red neuronal; las unidades de procesamiento que recogen información, la procesan y arrojan un valor; la organización del sistema de nodos para transmitir las señales desde los nodos de entrada a los nodos de salida y, por último, la función de aprendizaje a través de la cual el sistema se retroalimenta.

Se considera una red neuronal la ordenación secuencial de tres tipos básicos de nodos o capas: nodos de entrada, nodos de salida y nodos intermedios (capa oculta o escondida).

Los nodos de entrada se encargan de recibir los valores iniciales de los datos de cada caso para transmitirlos a la red. Los nodos de salida reciben entradas y calculan el valor de salida (no van a otro nodo). En casi todas las redes existe una tercera capa denominada oculta, Este conjunto de nodos utilizados por la red neuronal, junto con la

función de activación posibilita a las redes neuronales representar fácilmente las relaciones no lineales, que son muy problemáticas para las técnicas multivariantes.

Cuando se presenta un patrón de entrada X_p : $x_{p1},\ldots,x_{pi},\ldots x_{pN}$ se transmite a la red a través de los pesos w_{ji} desde la capa de entrada a la capa oculta. Las neuronas de esta capa transforman las señales a través de la función de activación proporcionando un valor de salida. Este valor se transmite a su vez a través de los pesos v_{kj} a la capa de salida donde aplicando de nuevo la función de activación obtenemos un valor de salida.



Vamos a suponer que la entrada total o neta de una neurona oculta j la expresamos como net_{pj} , entonces matemáticamente la podemos expresar de la siguiente manera:

$$net_{pj} = \sum_{i=1}^{N} w_{ji} x_{pi} + \theta_j$$

θ es el umbral de la neurona que se considera como un peso asociado a una neurona ficticia con valor de salida igual a 1.

El valor de salida de la neurona oculta j, b_{pj} lo obtenemos aplicando la función de activación f(.) sobre la entrada neta.

$$b_{pj} = f\left(net_{pj}\right)$$

La entrada neta que recibe una neurona de salida k la podemos expresar como:

$$net_{pk} = \sum_{j=1}^{L} v_{kj} b_{pj} + \theta_k$$

El valor de salida de la neurona k, y_{pk} es el siguiente:

$$y_{pk} = f\left(net_{pk}\right)$$

El objetivo que se persigue es hacer mínima la discrepancia o error entre la salida de la red y el valor real que presenta el usuario. La función que se pretende minimizar para cada patrón p viene dada por:

$$E_p = \frac{1}{2} \sum_{k=1}^{M} \left(d_{pk} - y_{pk}\right)^2$$

Donde d_{pk} es la salida presentada por la red de la neurona k ante la presentación del patrón p. La medida general del error es la suma de todos los errores para todos los patrones

$$E = \sum_{p=1}^{P} E_p$$

La técnica del gradiente decreciente [Rumelhart, Hinton y Williams, (1986)], conocida como algoritmo backpropagation por la modificación de los pesos se basa en el gradiente de E_p. Este gradiente toma la dirección que determina el incremento más rápido en el error. Así que el error puede reducirse ajustando cada peso en la siguiente dirección:

$$-\sum_{p=1}^{P} \frac{\partial E_p}{\partial w_{ji}}$$

3.3 Máquinas de vectores soporte.

Los fundamentos teóricos de las máquinas de vectores soporte (Support Vector Machines, SVM) fueron presentados en el año 1992 en la conferencia COLT (Computacional Learning Theory) por Boser, Guyon Y Vapnik (1992) y descritos posteriormente en diversos artículos por Cortes y Vapnik [Cortes y Vapnik (1995)]; Vapnik (1998) y (2000)] a partir de los trabajos sobre la teoría del aprendizaje estadístico.

Las máquinas de vectores soporte pertenecen a la familia de los clasificadores lineales dado que inducen hiperplanos o separadores lineales de muy alta dimensionalidad introducidos por funciones núcleo o kernel. Es decir, el enfoque de las SVM adopta un punto de vista no habitual, en vez de reducir la dimensión buscan una dimensión mayor en la cual los puntos puedan separarse linealmente.

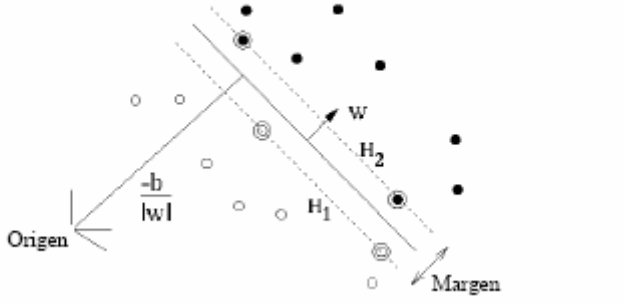
Máquinas de vectores soporte con margen máximo

La formulación matemática de las máquinas de vectores soporte se basa en el principio de minimización estructural del riesgo, que ha demostrado ser superior al principio de minimización del riesgo empírico utilizado en muchas de las redes convencionales.

Vamos a suponer que disponemos de una muestra S de N elementos del tipo (x_i , y_i) donde S={ (x_1 , z_1), (x_2 , z_2)..... (x_N , z_N)}.

Donde xi pertenece a un espacio de entrada y z_i toma los valores -1 y 1. x_i Є RP es el vector de variables para la observación i.

El conjunto de N datos es linealmente separable si es posible encontrar un vector w Є R^P que defina un plano que separe los puntos de ambas clases[1].



Un conjunto de observaciones se encuentra a un lado y verifica la siguiente ecuación:

$$w'x_i + b \leq -1$$

El otro conjunto de datos verifica

$$w'x_i + b \geq -1$$

Estas dos expresiones pueden escribirse como:

$$z_i\,(w'\,x_i + b) \geq 1 \text{ para } i = 1,2,.....N$$

Sea $f(x) = w'\,x_i + b$ el valor del hiperplano que separa óptimamente el conjunto de puntos.

[1] La ilustración que aparecen en este epígrafe ha sido tomadas de: Burges, Christopher J. C., "A tutorial on support vector machines for pattern recognition"

La distancia entre x_i y el hiperplano viene dada por la proyección del punto x_i en la w. Siendo w el vector ortogonal al plano.

La proyección se calcula a través de la siguiente expresión:

$$\frac{w' x_i}{\|w\|} \quad \|w\| \text{ es la norma en el espacio } R^P.$$

Maximizaremos la distancia de los puntos al plano maximizando la siguiente expresión

$$\frac{z_i (w' x_i + b)}{\|w\|}$$

Conseguiremos maximizar la anterior expresión siempre que el numerador sea positivo y el denominador lo más pequeño posible, lo que conduce a resolver un problema de programación cuadrática convexo bajo restricciones en forma de desigualdad lineal que se expresa de la siguiente forma:

$$\min \frac{1}{2}\|w\|^2$$

Sujeto a: $z\,(w'\,x_i + b) \geq 1$ para 1=1.....N.

Se define el margen funcional para un ejemplo (x , z) con respecto a una función f como el producto $z^*f(x)$, mientras que el margen normalizado geométrico de un hiperplano como 1/||w||. La solución de una SVM lineal con margen máximo es el hiperplano que maximiza el margen geométrico sujeto o restringido a que el margen funcional sea mayor o igual que 1.

El enfoque que utilizan los métodos clásicos es proyectar los datos sobre un espacio de dimensión menor y utilizar una función no lineal para discriminar, mientras que las SVM aplican una transformación de los datos de forma que los lleve a un espacio de dimensión mayor que p, y aplicar entonces una discriminación lineal como la anterior.

La forma habitual de resolver problemas de optimización con restricciones es utilizando la teoría desarrollada en 1797 por Lagrange y extendida después, para restricciones en forma de desigualdad, por Kuhn y Tucker en 1951. Su famoso teorema nos permite obtener una alternativa que se conoce como forma dual y que es equivalente a la forma primal pero que podemos expresar como una combinación lineal de los vectores de aprendizaje.

3.4 Redes bayesianas.

Las redes bayesianas se conocen en la literatura existente con otros nombres como redes causales o redes causales probabilísticas, redes de creencia, sistemas probabilísticas, sistemas expertos bayesianos o también como diagramas de influencia. Las redes bayesianas son métodos estadísticos que representan la incertidumbre a través de las relaciones de independencia condicional que se establecen entre ellas (Edwards, 1998). Este tipo de redes codifica la incertidumbre

asociada a cada variable por medio de probabilidades. Siguiendo a Kadie, Hovel y Horvitz (2001) afirman que una red bayesiana es un conjunto de variables, una estructura gráfica conectada a estas variables y un conjunto de distribuciones de probabilidad.

Estas redes probabilísticas automatizan el proceso de modelización probabilístico utilizando toda la expresividad de los grafos para representar las dependencias y de la teoría de la probabilidad para cuantificar esas relaciones. En esta unión se realiza de forma eficiente el aprendizaje automático como la inferencia con los datos y la información disponible

Una red bayesiana queda especificada formalmente por una dupla B = (G,Θ) donde G es un grafo dirigido acíclico (GDA) y Θ es el conjunto de distribuciones de probabilidad. Definimos un grafo como un par G= (V, E), donde V es un conjunto finito de vértices nodos o variables y E es un subconjunto del producto cartesiano V x V de pares ordenados de nodos que llamamos enlaces o aristas.

La redes bayesianas tienen la habilidad de codificar la causalidad entre las variables por lo que han sido muy utili**z**adas en el modelado o en la búsqueda automática de estructuras causales [López y García de la fuente; (2006)]. La potencia de las redes bayesianas está en su capacidad de codificar las dependencias/independencias relevantes considerando no sólo las dependencias marginales sino también las dependencias condicionales entre conjuntos de variables

La mayoría de los autores afirman que las redes bayesianas tienen dos dimensiones: una cuantitativa y otra cualitativa [Cowell, et alt., (1999); Garbolino y Taroni, (2002); Nadkarni y Shenoy, (2001), (2004); Martínez y Rodríguez, (2003)]

Los grafos definen un modelo probabilístico con las mismas dependencias utilizando una factorización mediante el producto de varias funciones de probabilidad condicionada:

$$p(x_i, x_2, \ldots\ldots x_n) = \prod_{i=1}^{n} p\left(x_i \,\middle|\, padres\left(x_i\right) \right)$$

$padres(x_i)$ son las variables predecesoras inmediatas de la variable x_i en la red, precisamente $p(x_i \,|\, padres(x_i))$ son los valores que se almacenan en el nodo que precede a la variable x_i

A través de la factorización las independencias del grafo son traducidas al modelo probabilístico de forma muy práctica.

4. Introducción a los métodos de combinación de modelos.

Una forma de conseguir una mayor precisión de las predicciones de nuestros modelos es acudir a los multiclasificadores. La combinación de las hipótesis de los multiclasificadores es una excelente forma de integrar la información de diferentes fuentes. Esta combinación de dos o más clasificadores, en general, como ya hemos afirmado en la introducción, proporciona estimaciones más robustas y eficientes que cuando se utiliza un único clasificador. También se utilizan porque resuelven el problema de sobreadaptación (overfitting) y es posible obtener buenos resultados con pocos datos.

Son múltiples los estudios que se han realizado con los métodos multiclasificadores, así que podemos conocerlos en la literatura existente con muchos nombres: métodos de ensamble, modelos múltiples, sistemas de múltiples clasificadores, combinación de clasificadores, integración de clasificadores, mezcla de expertos, comité de decisión, fusión de clasificadores de aprendizaje multimodelo.

Existen diferentes formas de combinar conjuntos de modelos. Dietterich (2000) estableció una clasificación atendiendo a diferentes criterios:

Manipulación de los datos entrenamiento: en estos procedimientos se construye un grupo de modelos mediante la repetición de k veces el mismo algoritmo de aprendizaje. Es importante el mecanismo de selección de los subconjuntos a partir de los datos de entrenamiento. Los métodos multiclasificadores más conocidos son: Bagging (Breiman 1996, Quinlan 1996a), Boosting, [Freund y Schapire 1996; Quinlan 1996b)] y Cross-validated Committes (Parmanto et al. 1996).

Manipulación de las variables de entrada: En esta técnica se altera el conjunto de atributos de entrada del algoritmo de aprendizaje. A esta familia de multiclasificadores pertenecen las técnicas Forest (Ho1998).

Métodos aleatorios: En estas técnicas se introducen componentes aleatorios en los proceso de aprendizaje con el objetivo de obtener diferentes multiclasificadores a partir de los mismos datos. Manipulación de los datos de salida.

Manipulación de los datos de salida. En problemas de clasificación se modifican las clases de los conjuntos de los ejemplo del conjunto de entren**a**miento.

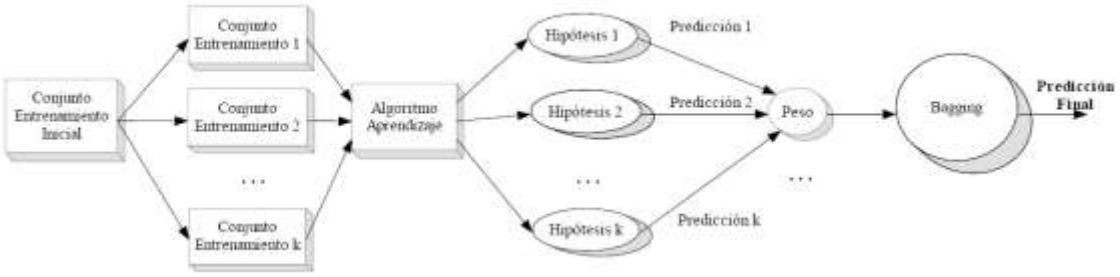
Existe una buena variedad de modelo de clasificación pero en este artículo sólo se describen brevemente los principales y más utilizados métodos multiclasificadores en minería de datos:

4.1 Bagging.

Este método propuesto por Breiman aúna las características del Bootstrapping y de la agregación incorporando los beneficios de ambos y dándole el nombre (Bootstrap AGGregatING). En este método se generan muestras aleatorias que serán los conjuntos de entrenamiento. Las muestras se generan a través de muestreo aleatorio con reemplazamiento. Cada subconjunto de entrenamiento aprende un modelo. Su principal utilidad es que reduce la varianza existente en la generación de cada modelo.

Para clasificar un ejemplo se predice la clase de ese ejemplo para cada clasificador y se clasifica la clase con mayor voto. Es decir, este método, a la hora de emitir una decisión, recurre a la decisión mayoritaria.

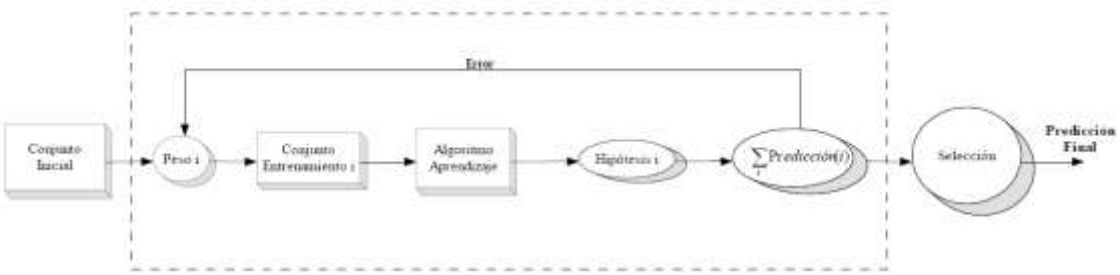
La arquitectura de este modelo se corresponde con el siguiente gráfico:



4.2 Boosting.

Este método fue propuesto por Freund y Schapire, R.E. El mecanismo que proponen sus autores está basado en la asignación de un peso a cada conjunto de entrenamiento. Cada vez que se itera se aprende un modelo que minimiza la suma de los pesos de aquellos ejemplos clasificados erróneamente. Los errores de cada iteración sirven para actualizar los pesos del conjunto de entrenamiento, incrementando el peso de los mal clasificados y reduciendo el peso en aquellos que han sido correctamente clasificados.

La estructura gráfica de este método es la siguiente:



La variante más conocida de estos algoritmos es AdaBoost y se encuentra implementada en WEKA.

4.3 Métodos de fusión.

Una vez construidos los modelos la predicción de nuevos casos se realiza mediante la fusión o combinación de las predicciones de cada modelo generado.

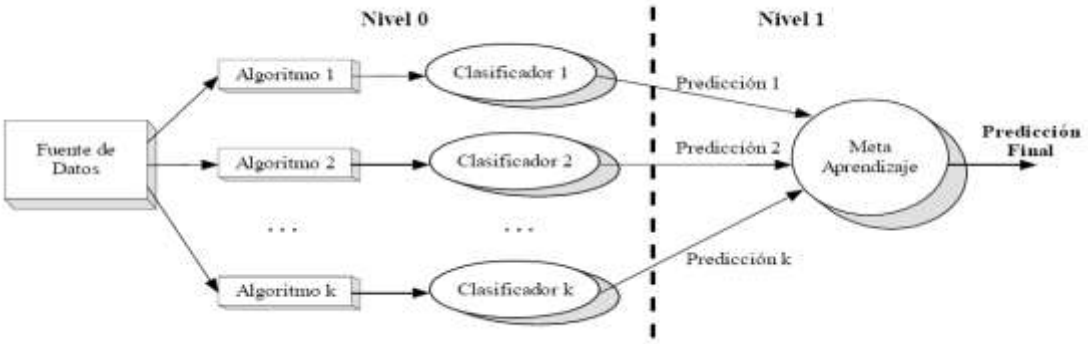
Siguiendo a Kuncheva (2002) vamos a suponemos que estamos trabajando con un multiclasificador que incluye m miembros o modelos y que se han definido varios métodos que nos permiten unificar los m vectores de probabilidad en un único vector α, entonces, algunas estrategias de fusión para extraer la clase predicha son las siguientes.

- Suma: $\alpha = \sum_{j=1}^{m} \nu_j$
- Media aritmética: $\alpha = \sum_{j=1}^{m} \frac{\nu_j}{m}$
- Producto: $\alpha = \prod_{j=i}^{m} \nu_j$
- Media geométrica: $\alpha = \sqrt[m]{\prod_{j=1}^{m} \nu_j}$
- Máximo: $\alpha = \max_{1 \le j \le m} (\nu_j)$
- Mínimo: $\alpha = \min_{1 \le j \le m} (\nu_j)$
- Mediana: $\alpha = mediana_{1 \le j \le m} (\nu_j)$

4.4 Métodos híbridos

4.4.1 Stacking

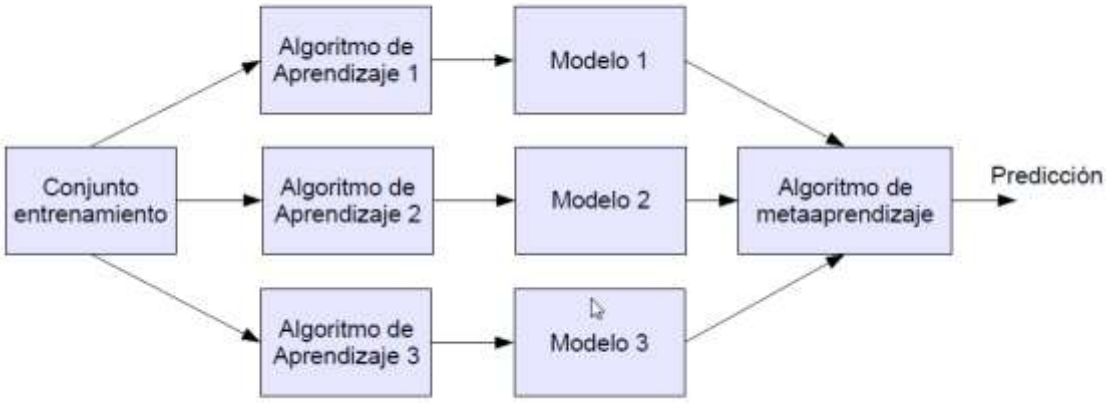
Este método combina múltiples clasificadores a través de diferentes algoritmos de aprendizaje. Los algoritmos de aprendizaje de la primera fase pueden ser árboles de decisión, redes neuronales, máquinas de vectores soporte, regresión logística, etcétera. En una segunda fase otro clasificador combina las salidas de los modelos de la fase anterior. La combinación de los clasificadores se realiza por mayoría. Este esquema funcionará bien cuando todos los modelos utilizados tienen una precisión aceptable.



En el ejemplo que se expone se combinan primeramente cinco modelos: Perceptrón Multicapa, Regresión logística, C4.5, máquinas de vectores soporte y Redes Bayesianas.

4.4.2 Cascading

Gama y Bradzil en el año 2000 presentan este método que nos permite mejorar las características de los árboles de decisión al incorporar nuevas particiones utilizando otros procedimientos de aprendizaje como hemos visto en el anterior multiclasificador. Cascading utilizan otros métodos de aprendizaje para crear nuevos atributos a través de redes neuronales, análisis discriminante, etc



5. Resultados obtenidos.

Los resultados que se ofrecen en este epígrafe se resumen en el cuadro nº 4 donde se detallan los resultados del porcentaje total de aciertos, desglosados para ambas clases y las medidas de evaluación de los catorce modelos que se han utilizado. Los métodos empleados han sido: la regresión logística como método paramétrico y siete modelos no paramétricos cuyos resultados son comparados con los que se obtienen a través de los seis métodos multiclasificadores.

Las instancias utilizadas han sido extraídas aplicando a la base de datos original el método del cubo a la clase dominante y el método de sobre muestreo denominado SMOTE a la clase minoritaria, descritos en las páginas anteriores. Al aplicar estos dos procedimientos se obtiene una base de datos que contiene 312 individuos de la clase SI (devuelve el crédito) y 310 de la clase NO (no pagan el crédito)

Se han aplicado dos redes bayesianas que buscan y optimizan la métrica bayesiana a través de los algoritmos K2 y TAN (Tree Augmented Naïve Bayes). La estructura bayesiana K2 obtiene resultados ligeramente más precisos que el algoritmo TAN. La fase de test se ha realizado con 26 registros seleccionados aleatoriamente de la base de datos.

El multiclasificador Stacking se configura con cinco modelos: perceptrón multicapa, red bayesiana con el algoritmo de búsqueda K2, regresión logística, máquinas de vectores soporte y el árbol de clasificación, C4.5.

En la fase de entrenamiento todos los modelos individuales utilizados son menos precisos que los multiclasificadores, si observamos el porcentaje de aciertos, el estadístico Kappa y el Área ROC. Entre estos el que más acierta es el Random Committee. Las redes bayesianas, especialmente cuando se utiliza el algoritmo de búsqueda K2, son las que mejores resultados arrojan, alcanzando un 84,9% de registros bien clasificados y un área de la curva ROC del 0.925. La regresión logística también ofrece, en esta etapa, buenos resultados.

Con los datos de test, que es la fase que realmente importa, dado que muchos de los métodos de minería de datos tienden a sobreajustarse, destacan los métodos bayesianos, que presentan los porcentajes más elevados de aciertos 84,6% y un área bajo la curva ROC del 0,881. Las redes neuronales de base radial al igual que el método bayesiano de Naïve Bayes presentan porcentajes de aciertos muy descompensados entre las clases. La regresión logística, en esta fase de test, no obtiene buenos resultados.

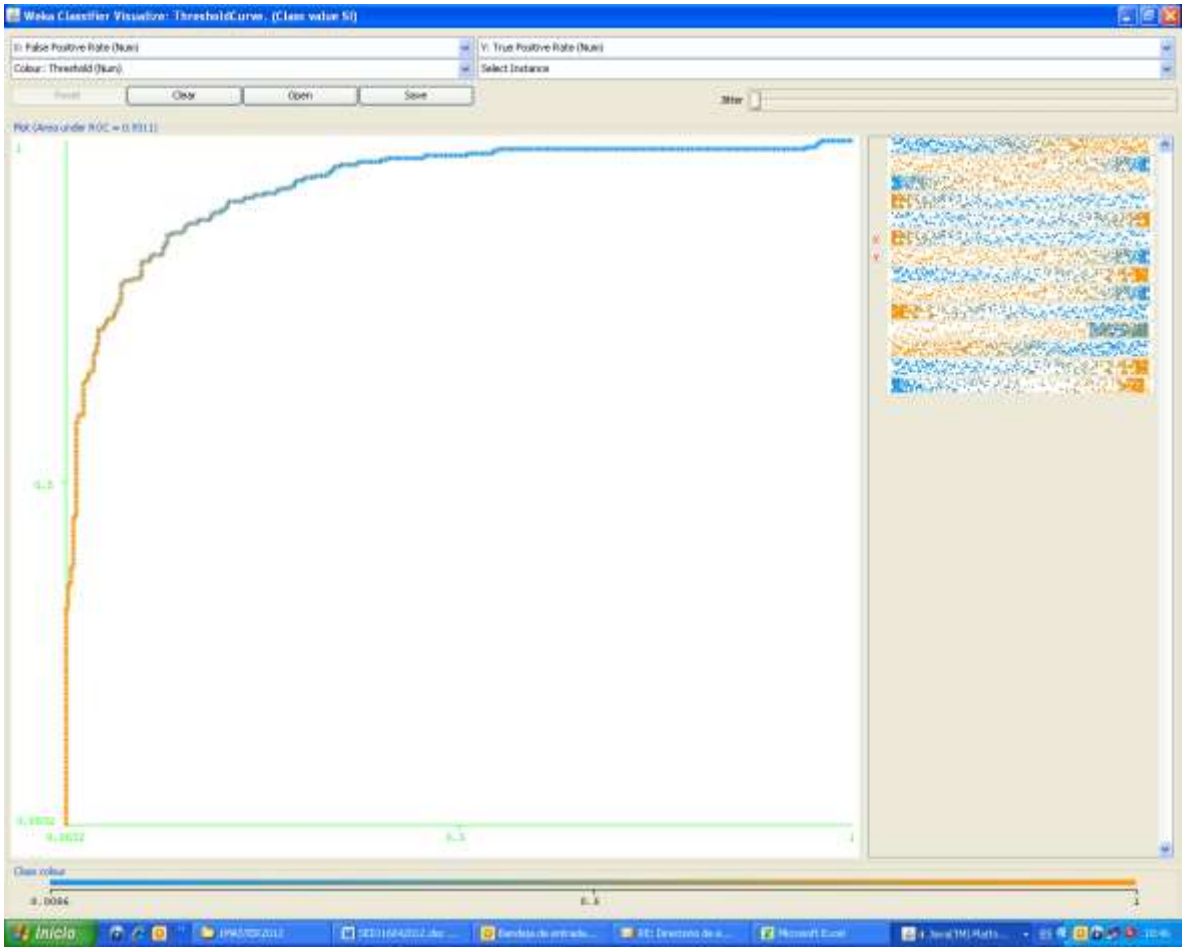
Entre los multiclasificadores, cuatro de ellos, presentan un 80,8% de aciertos. Bagging y Random Subspace, de las doce instancias de la clase minoritaria y más importante en términos de coste, predicen correctamente once de ellas, el 91,7% y, respecto a la otra clase, económicamente menos importante, solamente alcanzan el 71,4% de los registros correctamente clasificados.

Los métodos Stacking y Adaboost aciertan el 80,8% del total de registros y ofrecen un porcentaje más equilibrado de aciertos entre las clases.

CUADRO Nº 4. Fichero con SMOTE y MÉTODO DEL CUBO.

Técnica	CLASE SI (%)	CLASE NO (%)	TOTAL (%)	Estadístico KAPPA	AREA ROC
Fase de entrenamiento					
C 4.5	83,0	82,6	82,8	0,656	0,828
Maq. Vect. Soporte	82,7	82,6	82,6	0,653	0,826
Perceptrón Mult.	81,1	84,2	82,6	0,653	0,900
Redes Base Radial	75,6	82,3	78,9	0,579	0,852
NAÏVE BAYES	70,2	85,2	77,7	0,553	0,871
Red Bayesiana(TAN)	84,9	83,9	84,4	0,688	0,920
Red Bayesiana(K2)	84,6	85,2	84,9	0,698	0,925
Regresión Logística	84,0	84,8	84,4	0,688	0,916
Metaclasificadores					
Random Forest	83,0	90,0	86,5	0,730	0,924
ADABOOST	85,9	85,5	85,7	0,714	0,927
BAGGING	84,9	87,4	86,2	0,724	0,923
STAKING C (5 modelos)	83,7	87,4	85,5	0,711	0,931
Random Committee	85,9	89,4	87,6	0,752	0,942
RandomSubSpace	85,6	87,1	86,3	0,727	0,929
Fase de test					
C 4.5	71,4	66,7	69,2	0,381	0,664
Maq. Vect. Soporte	71,4	83,3	76,9	0,541	0,774
Perceptrón Mult.	71,4	75,0	73,1	0,462	0,726
Redes Base Radial	57,1	91,7	73,1	0,474	0,893
NAÏVE BAYES	50,0	91,7	69,2	0,402	0,881
Red Bayesiana(TAN)	78,6	91,7	84,6	0,694	0,851
Red Bayesiana(K2)	78,6	91,7	84,6	0,694	0,881
Regresión Logística	71,4	75,0	73,1	0,462	0,881
Random Forest	78,6	66,7	73,1	0,455	0,786
ADABOOST	78,6	83,3	80,8	0,615	0,863
BAGGING	71,4	91,7	80,8	0,620	0,857
STAKING C (5 modelos)	78,6	83,3	80,8	0,615	0,875
Random Committee	71,4	75,0	73,1	0,462	0,857
RandomSubSpace	71,4	91,7	80,8	0,620	0,875

Podemos visualizar la curva ROC resultante en la fase de entrenamiento para el multiclasificador Stacking, cuya salida origina el programa WEKA. El área bajo la curva, como se puede observar en la tabla de resultados, es del 0.931.



6. Conclusiones

Del análisis de este artículo se extraen, entre otras, las siguientes conclusiones:

- ✓ En general, podemos afirmar que para el credit scoring los métodos multiclasificadores obtienen mejores resultados que los algoritmos utilizados individualmente.
- ✓ La redes bayesianas alcanzan resultados excelentes tanto en la fase de entrenamiento como en la de test. Además, se convierten en excelentes modelos dado que pueden incorporar información de los expertos en el área de estudio y optimizan mejor el porcentaje de aciertos.
- ✓ Cuando las bases de datos están desbalanceadas las mejores opciones se experimentan cuando se equilibran las muestras. Se constata que existen muchas propuestas que intentar solucionar este problema sin que aún exista la solución ideal, sino que los resultados dependen de las características intrínsecas de los datos. La incorporación de una matriz de costes, como se observa en este artículo, produce unos resultados satisfactorios.
- ✓ Cuando el precio económico de la clasificación es diferente según las clases, incorporar la matriz de costes es muy conveniente. Los métodos Metacost y Cost Sensitive obtienen unos resultados muy aceptables.
- ✓ La selección de variables es una tarea imprescindible para buscar modelos más sencillos e interpretables.

7. Bibliografía.

Bonilla, M.;Olmeda, I.; Puertas, R.; (2003): Modelos paramétricos y no paramétricos en problemas de credit scoring. Revista española de financiación y contabilidad. Vol.XXXII, nº.118.

Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. (1992). A training algorithm for optimal margin classifiers. In David Haussler, editor, Proceedings of the 4th Workshop on Computational Learning Theory, pages 144–152, San Mateo, CA. ACM Press.

Breiman, L., Friedman, Jh., Olshen, R.A., & Stone, C.J. (1984). Classification and regression trees. Monterey, CA: Wadsworth & Brooks/Cole Advanced Book & Software.

Breiman, l. (1996): Bagging predictors. Machine Learning, Kluwer Academic Publishers, Boston, Manufactured in the Netherlands, vol. 24,2, 123-140

Breiman, L. (2001). Random Forests, random features. Berkeley: University of California.

Chawla, N.V.; Bowyer, K.W.; Hall. L.O. y kegelmeyer,W.P. (2002): "SMOTE: Synthetic Minority Over-Sampling Technique". Journal of Artificial Intelligence research, pp.321-357.

Cowell, R. G., A. P. David, S. L. Lauritzen, and D. J. Spiegelhalter (1999). Probabilistic Networks and Expert Systems. Springer-Verlag, New York.

Cohen, G.; Hilario, M.; Sax, H.; Hugonnet, S. Y Geissbuhler, A. (2006): "learning from imbalancing Data in Surveillance of Nosocomial Infection". Artificial Intelligence in Medicine, pp. 7-18.

Corinna Cortes and Vladimir Vapnik (1995). Support-vector networks. *Machine Learning*, 20(3):273-297.

Deville, J.-C. and Tillé, Y. (2004). Eficient balanced sampling: The cube method. Biometrika, 91:893_912.

Didzarevich, S.; Lizarraga, F.; Larrañaga, P.; Sierra, B., y Gallego, M.J. (1997): "Statistical and machine learning methods in the prediction of bankruptcy", en Sierra Molina, G. y

Domingos, P: MetaCost, 1999: A general method for making classifiers cost-sensitive. In: Fifth International Conference on Knowledge Discovery and Data Mining, 155-164.

Duda, H.; Stork (2001). Patternn Classifications, 2nd ed., Wiley.

Ferri, C.: Multiclasificadores en minería de datos. Dep. de Sistemes Informàtic i Computació, Universidad Politécnica de Valencia, Spain. Reunión Red Minería de Datos, Madrid, 2004.

Freund, Y.; Shapire, R.E. (1996). Experiments with a new boosting algoritm. Machine Learning: Proceedings of the 13th International Conference, July, 148-156.

Gama, J., Brazdil, P. (2000): Cascade Generalization. Machine Learning, Kluwer Academic PUblishers, Boston, Manufactured in the Netherlands, vol. 41, 3, 315-343.

Garbolino, P., y Taroni, F. (2002). Evaluation of scientific evidence using bayesian networks. Forensic Science International, 125, 149-155.

Han, H.; Wang, W. y Mao, B. (2005): "Borderline-SMOTE: anew Over-Sampling Method in Imbalanced Data Sets Learning". En: D.-S. Huanng; X.-P.Zhzng y G.-B. Huang (Eds.), ICICS, volumen 3644 de LNCS, pp. 878-887.

Hernández Orallo, J.,Ramírez Quintan, M.J. y Ferri Ramírez, C. (2004): Introdución a la minería de datos. Pearson – Prentice Hall.

Ho, T.K., 1998. The random subspace method for constructing decision forests. IEEE. Trans. Pattern Anal. Machine Intell. 20 (8), 832–844.

Holland. J.H. (1975). Adaptation in Natural and Arti_cial Systems. The University of Michigan Press (The MIT Press, London, 1992).

Japkowicz, N. (2001): "Concept-Learning in the Presence of Between-Class and Within-Class Imbalances". En: E. Stroulia y S. Matwin (Eds.), Canadian Conference on AI, volume 2056 de LNCS, pp. 67-77.

Japkowicz (2004), Estabrooks, A., Jo, T. "A Multiple Resampling Method for Learning from Imbalances Data Sets" , Estabrooks, A., Jo, T. and Japkowicz, N., Computational Intelligence, Volume 20, Number 1, February 2004, pp.18-36.

Kubat, M. Y Matwin, S. (1997): "Addressing the Course of Imbalanced Training Sets: One-Sided Selection". En: D.H.Fisher (Ed.), ICML, pp. 179-186.

Kuncheva, L. y Jain. L.C. (1999): "Nearest neighbor classifier: Simultaneous editing and feature selection" pattern Recognition Letters, pp.1149-1156.

Kuncheva, L.I. (2002): A Theorical study on six classifier fusion strategies. IEEE Transaction on PAMI, 24 (2), pp. 281-286.

Laurikkala, J.(2002): "Instance-based data reduction for improved identification of difficult small classes". Intelligent Data Analysis, pp.311-322.

López, J.-García, J.-De la Fuente, L.(2006): Modelado causal con redes bayesianas. Actas de las XXVII Jornadas de Automática, 198–202.

Marczyk, Adam. 2004. Genetic algorithms and evolutionary computation. The Talk, Origins Archive. 23 Apr. 2004. 7 Oct. 2006.

Nadkarni, S., y Shenoy, P. P. (2001). A bayesian network approach to making inferences in causal maps. European Journal of Operational Research, 128, 479-498.

Nadkarni, S., y Shenoy, P. P. (2004). A causal mapping approach to constructing bayesian networks. Decision Support Systems, 38, 259-281.

PARMANTO B, PAUL W MUNRO & HOWARD R “Reducing Variance of Committee Prediction with Resampling Techniques. Connection Science. Volume 8, Issue 3-4, 1996

Provost, F. (2003): “Machine learning from imbalanced data sets 101 (Extended Abstract)”. En: AAAI: Workshop on Learning with Imbalanced Data Sets.

Quinlan, J.R. (1993): C4.5: Programs for machine learning, Morgan Kaufmann Publishers, Inc., California (USA).

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations. David E. Rumelhart, James L. McClelland and the PDP Research Group. Cambridge, MA: MIT Press, pp. 318--362.

Vapnik, V. (1998). Statistical Learning Theory. John Wiley and Sons, Inc., New York, 1998.

Vapnik. (2000). "The Nature of Statistical Learning Theory (2nd Edition)." Springer.

Wang, J.; Xu, M.; Wang, H. Y Zhang, J. (2006): “Clasification of Imbalanced Data by Using the SMOTE Algorithm and locally Linear Embedding”. En: ICSP, volume 3, pp. 16-20.

Wilson, D. L. (1972), Asymptotic properties of nearest neighbor rules using edited data, IEEE Transactions on Systems, Man and Cybernetics . IEEE Computer Society Press, Los Alamos.

Zhang, J. Y Mani, I. (2003).: “kNN approach to unbalanced data distributions: a case study involving information extraction”. En ICML: Workshop on Learning from Imbalanced Dataset II.