Aprendizaje mediante la hibridación de técnicas heurísticas y estadísticas de optimización en regresión logística binaria

César Hervás[1], Francisco J. Martínez[2], Alfonso C. Martínez[2], Pedro A. Gutierrez[1], Juan C. Fernández[1]

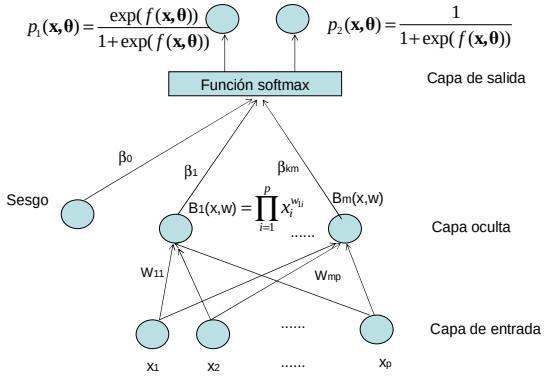
Resumen-- **Este trabajo aborda la resolución de problemas de clasificación binaria utilizando una metodología híbrida que combina la regresión logística y modelos evolutivos de redes neuronales de unidades producto. Para estimar los coeficientes del modelo lo haremos en dos etapas, en la primera aprendemos los exponentes de las funciones unidades producto, entrenando los modelos de redes neuronales mediante computación evolutiva y una vez estimados el número de funciones potenciales y los exponentes de estas funciones, se aplica el método de máxima verosimilitud al espacio de características formado por las covariables iniciales junto con las nuevas funciones de base obtenidas al entrenar los modelos de unidades producto. Esta metodología híbrida en el diseño del modelo y en la estimación de los coeficientes se aplica a cuatro bases de datos de referencia. Los resultados obtenidos con este modelo híbrido mejoran, en todas las bases de datos, los obtenidos con una regresión logística en cuanto a porcentaje de patrones bien clasificados sobre el conjunto de generalización, con un número de coeficientes del modelo muy ajustado de forma tal que mejore la interpretabilidad de los mejores modelos propuestos.**

Palabras clave-- **Regresión logística, Redes Neuronales de Unidades Producto, Algoritmo Evolutivo, Clasificación.**

I. INTRODUCCIÓN

Existen muchos campos de investigación como medicina, microbiología, epidemiología y muchos otros, donde es muy importante predecir el resultado de una variable de respuesta binaria, o de forma similar obtener la probabilidad de éxito en una variable aleatoria de Bernouilli, en función de la relación que tiene esta variable con un conjunto de variables explicativas o covariables. De esta forma, si lo consideramos como un problema de aprendizaje supervisado de clasificación binaria, la meta es aprender como distinguir ejemplos que pertenecen a una de entre dos clases (caracterizadas por los sucesos Y=1, e Y=0) en función de los valores que toman k variables predictoras o covariables $X_1, X_2, ..., X_k$.

El modelo de regresión logística ha sido ampliamente empleado como método de regresión en estadística desde hace muchos años y, recientemente, ha recibido una gran aceptación en la comunidad de "machine learning" debido a su cercana relación con las nuevas teorías asociadas a las máquinas de soporte vectorial SVM [1] y a las técnicas de remuestreo iterativo como puede ser AdaBoost [2]. La regresión logística es un modelo de regresión lineal generalizada donde se trata de predecir las probabilidades a posteriori de pertenencia de cada uno de los patrones de un conjunto de entrenamiento a uno de los dos valores que toma la variable dependiente mediante relaciones lineales con las variables predictoras [3]. Este tipo de modelos de regresión, donde se hace lineal el modelo mediante transformaciones de la variable dependiente, se aplica con mucha frecuencia a problemas donde la variable de respuesta es dicotómica, de forma tal que se le puede asignar los sucesos éxito y fracaso como se hace por ejemplo en [4]. Por otra parte, un modelo de regresión logística se puede representar de forma equivalente a como se representa una estructura de grafo de tipo perceptrón, con una función de activación logística, representando un esquema de red neuronal lo más sencilla posible, como se observa en la Figura 1.



Además, en los últimos años se esta haciendo hincapié en la comparación de las nuevas técnicas y algoritmos de clasificación basados en modelos de redes neuronales y de árboles de decisión con modelos clásicos estadísticos como los análisis discriminante lineal y cuadrático y los modelos de regresión logística estándar o con restricciones [5],[6].

[1]Departamento de Computación Universidad de Córdoba 14071-Córdoba-España.
chervas@uco.es
[2] Facultad de Ciencias Económicas y Empresariales. ETEA.{fjmestud,acme}@etea.com

El método habitual de estimación de los coeficientes de un modelo lineal de regresión logística es el de máxima verosimilitud, dado que la distribución de probabilidad de la variable dependiente es una distribución de Bernouilli de parámetro p, la probabilidad de éxito, asociada al valor Y= 1 de la variable dependiente. Para obtener el óptimo de la función de verosimilitud se utiliza habitualmente un método de optimización local basado en un algoritmo iterativo de tipo Newton-Raphson o de mínimos cuadrados con reasignación de pesos (IRLS) [7].

Desde un punto de vista formal, la regresión logística es un procedimiento sencillo y útil, pero no siempre se verifica que las probabilidades de pertenencia a cada clase transformadas mediante una transformación logarítmica presenten una relación lineal, causa-efecto, sobre las covariables.

Una aproximación para evitar estas dificultades es aumentar o reemplazar el vector de entradas x añadiéndole variables adicionales, o funciones de base, las cuales son transformaciones del vector x, para a continuación utilizar el modelo lineal en este nuevo espacio de características de entrada. La ventaja de esta metodología es que una vez que se determine el número y la estructura de estas funciones base, el modelo es lineal en estas nuevas variables y podemos estimar los coeficientes del modelo mediante el procedimiento probabilístico habitual de maximizar la función de verosimilitud.

Existen diferentes tipos de funciones de base, entre ellas destacamos las sigmoides que dan lugar a los modelos de redes neuronales multicapa, MLP, [8], las funciones de tipo “kernel” o ventana donde las funciones más habituales son las gaussianas y que dan lugar a los modelos de base radial, RBF, y aquellas que dan lugar a los modelos de aprendizaje basado en proyecciones “projection pursuit learning” [9], los modelos lineales generalizados [10] y los métodos adaptativos multivariantes de tipo “spline” como MARS [11], o la muy reciente de utilizar funciones recíprocas sigmoides [12] pero existen dificultades a la hora de su utilización, debido a la complejidad de estos modelos derivada del diseño de la tipología y del número de funciones de base necesarios para un problema concreto de clasificación.

Para introducir no linealidad en el modelo y poder abordar otras metodologías de clasificación utilizando técnicas heurísticas y técnicas estadísticas de optimización, proponemos en este trabajo un modelo de regresión logística basado en la hibridación del modelo lineal en las covariables iniciales y en las funciones de base (no lineales) de tipo producto. Estas funciones que introducimos en el modelo lineal, expresarán las posibles interacciones significativas existentes entre las covariables. Los exponentes de estas unidades producto no son fijos sino que pertenecen al conjunto de los números reales. De esta forma, la parte no lineal del modelo propuesto se corresponde con una clase especial de redes neuronales, llamadas redes neuronales de unidades producto, PUNN, [13], siendo una alternativa a las redes neuronales de unidades sigmoides o redes MLP.

Desafortunadamente en nuestra aproximación no podemos garantizar la no existencia de óptimos globales en la superficie de log-verosimilitud, como ocurre en los modelos estándar de regresión logística. En efecto, las superficies de error asociadas a las redes PUNN son extremadamente complejas con numerosos óptimos locales y superficies planas. Esto es debido a que pequeños cambios en los exponentes de las variables producen grandes cambios en la superficie de error.

El aprendizaje de los coeficientes y del número de funciones de base se lleva a cabo en varias etapas. En la primera, se utiliza un algoritmo evolutivo (EA) para diseñar la estructura: número de funciones de base, y el aprendizaje de los pesos asociados a los exponentes de las covariables en el modelo PUNN. La complejidad de la superficie de error del modelo propuesto justifica la utilización de un algoritmo evolutivo como parte del procedimiento de estimación de los coeficientes del modelo. Es bien conocido que los algoritmos evolutivos son eficientes a la hora de explorar el espacio de búsqueda; sin embargo, su rendimiento es pobre a la hora de encontrar con precisión una solución óptima en la región donde el algoritmo converge.

Para poder mejorar la estimación de los coeficientes de regresión del modelo lineal de regresión logística debido a la falta de precisión del algoritmo evolutivo, utilizamos en una segunda etapa, un algoritmo de optimización de la función de máxima verosimilitud asociada al nuevo modelo de regresión logística. Si precisamos más, una vez que tenemos el número y los exponentes de las funciones de base mediante el algoritmo evolutivo, podemos considerar el modelo como lineal tanto en estas nuevas variables como en las covariables iniciales y podemos obtener los estimadores de los coeficientes de regresión mediante un procedimiento de máxima verosimilitud asociado al modelo estándar de regresión logística. Por último, aplicamos un método de selección de covariables en función de su capacidad para explicar la variable de respuesta. Controlando de esta forma el número de coeficientes del modelo final podremos disminuir el riesgo de construir un modelo complejo que sobreaprenda los datos del conjunto de entrenamiento, con la consiguiente pérdida de capacidad de generalización sobre los datos del conjunto de test.

Para evaluar el rendimiento de nuestra metodología utilizamos cuatro bases de datos con dos posibilidades de clasificación de los patrones,

bases de datos que han sido tomadas del repositorio de la Universidad de California en Irving, UCI [14]. Los resultados empíricos muestran que tanto la metodología como el modelo propuesto es muy prometedor tanto en su capacidad de precisión en la clasificación, como en su simplicidad, dada por el relativamente pequeño número de coeficientes y de funciones de base utilizadas en los mejores modelos propuestos como clasificadores.

Es interesante apuntar que el modelo propuesto mejora los resultados obtenidos con el modelo de regresión logística utilizando las covariables iniciales (LR). De esta forma el modelo híbrido (LRLPU) determina un buen balance entre la estructura lineal y no lineal del modelo.

El trabajo está organizado de forma tal que en la sección II se hace una revisión de trabajos relacionados con el tema que nos ocupa. En la sección III hacemos una breve introducción a la regresión logística para, a continuación, presentar nuestro modelo en profundidad. En la sección IV describimos el procedimiento de estimación de los coeficientes del modelo basado en dos etapas con un método diferente de optimización en cada una de ellas. En las secciones V y VI mostramos los resultados para las bases de datos de prueba utilizados para analizar el rendimiento de nuestro algoritmo y en la sección VI presentamos las conclusiones y nuevas propuestas de futuro en este campo.

II. TRABAJOS RELACIONADOS

En esta sección damos una breve reseña acerca de los diferentes métodos que utilizan funciones de base para ir más allá de la linealidad del modelo, entre ellos citaremos trabajos recientes que muestran la proximidad existente entre modelos de regresión logística y métodos de "machine learning".

Los modelos aditivos generalizados [10] comprenden métodos estadísticos automáticos y flexibles que se utilizan para identificar y caracterizar los efectos de la regresión no lineal. Para un problema de clasificación en dos clases, el modelo de regresión logística es un ejemplo de modelo aditivo generalizado donde se reemplaza cada término lineal por una forma funcional más general que aproxima funciones multidimensionales mediante la suma de funciones unidimensionales.

Uno de los modelos más populares de regresión no lineal es la regresión de tipo "spline" adaptativa multivariante (MARS) [11], donde las funciones base vienen determinadas por el producto de algún número de funciones unidimensionales de tipo "spline". Las funciones base se añaden incrementalmente durante el aprendizaje, utilizando una técnica constructiva de selección secuencial.

Bajo otro punto de vista, la regresión logística ha ganado popularidad recientemente en la comunidad de investigadores en "machine learning" debido a su relación con técnicas bien conocidas como las máquinas de soporte vectorial, SVM, [1], las técnicas de árboles de decisión con remuestreo como AdaBoost [2] y las redes neuronales artificiales [6], [5]. Vapnik [1] compara la regresión logística, LR, con SVM a través de la minimización de la función de pérdida, y muestra que la función de pérdida del modelo LR puede ser muy bien aproximada por la función de pérdida SVM con múltiples grupos (SVMn).

III. REGRESIÓN LOGÍSTICA CON FUNCIONES DE BASE DE TIPO PRODUCTO

Sea Y una variable aleatoria de Bernouilli asociada a un problema de clasificación en dos clases, de forma tal que codificamos como Y= 1 la pertenencia a la primera clase C_1, con una probabilidad p condicionada a los valores de las covariables y como Y= 0 la pertenencia a la segunda clase, C_2, y sea $X_1,X_2,\ldots,X_k$ un conjunto de k variables independientes. Sea $D=\{(\mathbf{x}_l, y_l): l=1,2,...,n_T\}$ un conjunto de individuos o patrones de tamaño que forman el conjunto de entrenamiento del clasificador. El modelo de regresión logística [15], [16], [16] es una técnica habitual en estadística en la cual la probabilidad p de pertenencia a la primer clase está relacionada con una serie de valores del conjunto de entrenamiento de las variables explicativas o covariables $\mathbf{x}=(1, x_1,\ldots,x_k)$ en la forma

$$\text{logit}(p)=\ln\left(\frac{p}{1-p}\right)=f(\mathbf{x},\boldsymbol{\beta})=\boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}=\sum_{i=0}^{k}\beta_i x_i \quad (1)$$

donde $\boldsymbol{\beta}=(\beta_0,\beta_1,\beta_2,...,\beta_k)$ son los coeficientes del modelo. Se define al cociente $\frac{p}{1-p}$ como la razón de probabilidades o "odds-ratio" y a la expresión (1) como el "log-odds" o transformación "logit".

Un sencillo cálculo basado en la ecuación (1) muestra la probabilidad de éxito como una función no lineal de las covariables y viene dada por:

$$p(\mathbf{x};\boldsymbol{\beta})=\frac{e^{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}}}{1+e^{\boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}}} \quad (2)$$

Si conocemos la función de probabilidad condicional (2), podemos construir una regla de decisión óptima Bayesiana en la forma

$$r(\mathbf{x})=sign\left\{\ln\left(\frac{p}{1-p}\right)\right\} \quad (3)$$

Esto es, si $r(x)>0$ entonces $x\in C_1$; y si $r(x)<0$ entonces $x\in C_2$. La frontera de decisión, si no tenemos probabilidades a priori de pertenencia a cada clase, es el conjunto de puntos para los cuales $\log(p/(1-p))=0$, esto es, el hiperplano de regresión definido por la ecuación $\boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}=0$.

Hay que observar que la regresión logística no solo construye una regla de decisión sino que encuentra una función de forma tal que para cualquier patrón con un vector de características o covariables x estima la probabilidad p de que dicho patrón pertenezca a la primera clase.

Sea $D = \{(\mathbf{x}_l, y_l) : l = 1, 2, ..., n_T\}$ el conjunto de datos de entrenamiento, donde el número de patrones es n_T. Aquí, suponemos que la muestra de entrenamiento es una realización de un conjunto de variables aleatorias independientes e idénticamente distribuidas. Los coeficientes de regresión $\boldsymbol{\beta}$, se estiman a partir de los datos ese conjunto y son directamente interpretables como proporciones de logaritmo de cocientes de probabilidad, en términos de la forma $\exp(\beta_1)$, o como proporciones de cocientes de probabilidades. El método de estimación de estos coeficientes se basa habitualmente en el método de máxima verosimilitud, esto es, en la maximización del logaritmo de la función de verosimilitud; siendo este logaritmo de la función de verosimilitud para n_T observaciones:

$$l(\boldsymbol{\beta}) = \sum_{l=1}^{n_T} \{ y_l \log p(\mathbf{x}_l; \boldsymbol{\beta}) + (1 - y_l) \log(1 - p(\mathbf{x}_l; \boldsymbol{\beta})\} = \sum_{l=1}^{n_T} \left\{ y_l \boldsymbol{\beta}^{\mathbf{T}} \mathbf{x}_l - \log(1 + e^{\boldsymbol{\beta}^{\mathbf{T}} \mathbf{x}_l}) \right\} \quad (4)$$

La matriz Hessiana asociada al procedimiento de maximización de $l(\boldsymbol{\beta})$ es semidefinida negativa [17], lo que implica que dicha función es cóncava sobre el parámetro $\boldsymbol{\beta}$. La concavidad, junto con el hecho de que el vector de parámetros $\boldsymbol{\beta}$ varíe libremente sobre un conjunto convexo garantiza que no existe un máximo local sobre la superficie del logaritmo de la función de verosimilitud en un modelo de regresión logística. Las condiciones bajo las cuales existe un máximo global y los estimadores obtenidos por el método de máxima verosimilitud no divergen se discuten en [17]. La estimación de los parámetros $\boldsymbol{\beta}$ habitualmente se realiza mediante un procedimiento iterativo como el algoritmo de Newton-Raphson o el algoritmo iterativo de mínimos cuadrados con reasignación de pesos (IRLS) [7]. Por lo general el algoritmo no converge dado que el logaritmo de la función de verosimilitud es cóncavo. En los casos poco probables de que la log-función de verosimilitud decrezca, el tamaño de la etapa garantiza la convergencia.

Desde un punto de vista formal, el modelo de regresión logística es muy sencillo, pero a veces los resultados de clasificación son pobres cuando se aplica a un problema real de clasificación, especialmente si queremos hacer predicciones de las probabilidades de pertenencia a una clase en los extremos del dominio de los datos del conjunto de entrenamiento. Por esta razón es conveniente a veces, aumentar o reemplazar el vector de covariables de entrada $\mathbf{x}$ mediante variables adicionales, las cuales son transformaciones de $\mathbf{x}$, y utilizar el modelo de regresión logística sobre este nuevo espacio de características de entrada.

Para salvar el efecto de la no linealidad de las covariables y reducir el error en los bordes del dominio del conjunto de datos, proponemos un modelo de regresión logística basado en la hibridación de un modelo estándar de regresión logística y un modelo de red neuronal de unidades producto, PUNN, introduciendo un término no lineal en el modelo mediante la construcción de funciones de base obtenidas mediante la multiplicación de potencias de las covariables iniciales, las cuales expresan las posibles fuertes interconexiones existentes entre las covariables. La expresión general del modelo es la siguiente:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \alpha_0 + \sum_{i=1}^{k} \alpha_i x_i + \sum_{j=1}^{m} \beta_j \prod_{i=1}^{k} x^{w_{ji}}$$

que en notación matricial es de la forma:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \boldsymbol{\alpha}^{\mathbf{T}} \mathbf{x} + \boldsymbol{\beta}^{\mathbf{T}} \mathbf{B}(\mathbf{x}, \mathbf{W})$$

donde las funciones base son

$$\mathbf{B}(\mathbf{x}, \mathbf{W}) = \{ B_1(\mathbf{x}, \mathbf{w}_1), B_2(\mathbf{x}, \mathbf{w}_2), ..., B_m(\mathbf{x}, \mathbf{w}_m) \},$$

con $B_j(\mathbf{x}, \mathbf{w}_j) = \prod_{i=1}^{k} x_i^{w_{ji}}$, $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{W})$,

$\boldsymbol{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_k)$, $\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_k)$ y $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_m)$, con $\mathbf{w}_j = (w_{j1}, w_{j2}, ..., w_{jk})$, $w_{ij} \in \mathbb{R}$.

De esta manera, la nueva función de probabilidad condicional es:

$$p(\mathbf{x}; \boldsymbol{\theta}) = \frac{e^{f(\mathbf{x}, \boldsymbol{\theta})}}{1 + e^{f(\mathbf{x}, \boldsymbol{\theta})}} \quad (4)$$

y la transformación logit:

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = f(\mathbf{x}, \boldsymbol{\theta}) \quad (5)$$

En este caso los bordes de decisión son funciones no lineales y están definidas por la hipersuperficie $f(\mathbf{x}, \boldsymbol{\theta}) = 0$ en el espacio $\mathbb{R}^k$. Dependiendo del parámetro $\boldsymbol{\theta}$, la hipersuperficie puede incluso ser no conexa. La representación del modelo en grafo se observa en la figura 1.

La parte no lineal de $f(\mathbf{x}, \boldsymbol{\theta})$, se corresponde con una clase especial de modelos de redes neuronales con activación hacia delante, esto es, las redes neuronales de unidades producto (PUNN), introducidas por Durbin y Rumelhart [13]. Este tipo de redes son una alternativa a los modelos estándar donde se utilizan funciones de base sigmoide y se basan en considerar en los nodos ocultos funciones de transferencia multiplicativas en vez de aditivas. Esta clase de redes neuronales multiplicativas

comprende otros tipos de redes como las redes sigma-pi y las redes de unidades producto. En contraste con las unidades sigma-pi, en las unidades producto los exponentes no son fijos y pueden tomar valores reales. Las ventajas de las redes neuronales basadas en las unidades producto son que aumentan la capacidad de información y la habilidad para formar combinaciones de las covariables de entrada de un alto orden, además este tipo de funciones son aproximadores universales [18], [18] y es posible obtener límites superiores de la dimensión de Vapnik-Chervonenkis, VC, de las redes de unidades producto, similares a las obtenidas mediante redes neuronales de unidades sigmoides [19].

A pesar de estas claras ventajas, las redes basadas en unidades producto tienen una seria dificultad. Esta dificultad consiste en que su entrenamiento es más difícil que el entrenamiento de las redes MLP [13]. La principal razón para que exista esta dificultad es que pequeños cambios en los exponentes de las variables pueden producir grandes cambios en la superficie de error. Debido a ello, la superficie de error de las redes de unidades producto tiene más mínimos locales y es más probable que el algoritmo de búsqueda se quede atrapado en uno de ellos [20], [21].

En este entorno de trabajo, las unidades producto utilizadas en este trabajo tienen una capa de entrada con un nodo para cada covariable, una capa oculta con varios nodos, y una capa de salida con un único nodo. No existen conexiones entre los nodos de una misma capa y no existen tampoco entre las capas de entrada y de salida.

La red tiene *k* variables de entrada o covariables que representan las variables independientes del modelo, *m* nodos en la capa oculta y un nodo en la capa de salida. La función de activación del nodo j-ésimo de la capa oculta viene dada por $B_j(\mathbf{x},\mathbf{w_j}) = \prod_{i=1}^{k} x_i^{w_{ji}}$ donde w_{ji} es el peso de la conexión existente entre el nodo de entrada i y el nodo oculto j. La función de activación de la capa de salida viene dada por $f(\mathbf{x},\boldsymbol{\theta}) = \beta_0 + \sum_{j=1}^{m} \beta_j \prod_{i=1}^{k} x_i^{w_{ji}}$, donde β_j es el peso de la conexión entre el nodo oculto j y el nodo de salida. La función de transferencia de todos los nodos de las capas oculta y de salida es la función identidad.

Por otra parte, y retomando el modelo propuesto en (5) el logaritmo de la función de verosimilitud para n_T observaciones es:

$$l(\boldsymbol{\theta}) = \sum_{l=1}^{n_T} \left\{ y_l f(\mathbf{x}_l,\boldsymbol{\theta}) - \log(1+e^{f(\mathbf{x}_l,\boldsymbol{\theta})}) \right\} \qquad (6)$$

La naturaleza y las propiedades de la función $f(\mathbf{x},\boldsymbol{\theta})$ implica que la matriz Hessiana asociada a la maximización de (6) es en general indefinida y que puede tener máximos locales, y por tanto, es probable que el algoritmo se quede atrapado en ellos. De esta forma, nuestra aproximación no puede garantizar que se alcance el máximo global en la superficie de verosimilitud. Para salvar este problema utilizamos un algoritmo evolutivo como parte del proceso de estimación de los coeficientes del modelo y también como un método para determinar el número, m, de unidades de base óptimo. En la siguiente sección presentamos el procedimiento de obtención de los estimadores $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{W}})$ de los coeficientes del modelo.

III. Aprendizaje de los parámetros

La propuesta se basa en la combinación de un algoritmo evolutivo (explorador global) y de un procedimiento de optimización local (explotador local) llevado a cabo mediante un procedimiento de maximización de la función de verosimilitud asociada a un modelo de regresión logística. En una primera etapa, se aplica un algoritmo evolutivo (EA) para diseñar la estructura y el entrenamiento de los exponentes de las funciones de base de redes neuronales de unidades producto. El proceso evolutivo determina el número *m* de funciones multiplicativas de potencias de las covariables iniciales del problema, así como los vectores $\mathbf{w_j}$, correspondientes a los exponentes de cada una de estas *m* funciones. Una vez que tenemos determinadas las funciones base mediante el algoritmo evolutivo,

$B(\mathbf{x},\mathbf{W}) = \{B_1(\mathbf{x},\mathbf{w}_1), B_2(\mathbf{x},\mathbf{w}_2), ..., B_m(\mathbf{x},\mathbf{w}_m)\}$,

consideramos un aumento del espacio de entrada añadiendo a las covariables iniciales estas nuevas variables, transformaciones no lineales de las covariables. De esta forma, el nuevo modelo es lineal en este nuevo espacio de variables, y los restantes coeficientes del modelo $\boldsymbol{\alpha}$ y $\boldsymbol{\beta}$ se calculan mediante un procedimiento de máxima verosimilitud condicional asociado a la regresión logística estándar. Por último, utilizamos un procedimiento por etapas para eliminar variables de forma secuencial del modelo completo, hasta que ninguna nueva eliminación mejore su capacidad de clasificación sobre el conjunto de test.

A continuación mostramos con más detalle los diferentes aspectos de esta metodología.

Etapa 1. Aplicamos un algoritmo evolutivo para encontrar las funciones base de unidades producto, $B(\mathbf{x},\hat{\mathbf{w}}) = \{B_1(\mathbf{x},\hat{\mathbf{w}}_1), B_2(\mathbf{x},\hat{\mathbf{w}}_2), ..., B_m(\mathbf{x},\hat{\mathbf{w}}_m)\}$, correspondientes a la parte no lineal de la función $f(\mathbf{x},\boldsymbol{\theta})$. Debemos determinar el número de funciones m y la matriz de pesos formada por los exponentes estimados de las funciones potenciales $\hat{\mathbf{W}} = (\hat{\mathbf{w}}_1, \hat{\mathbf{w}}_2, ..., \hat{\mathbf{w}}_m)$. Entre los diferentes paradigmas de la Computación Evolutiva,

hemos elegido la Programación Evolutiva (EP) debido al hecho de que estamos evolucionando modelos cuya representación en grafo está asociada a un modelo de red neuronal artificial con números reales como pesos. Nuestro algoritmo evolutivo para diseñar la arquitectura del modelo y la estimación de los exponentes reales tiene puntos en común con otros algoritmos evolutivos referenciados en la bibliografía [22],[23], y puede verse con detalle en [22], donde la única diferencia planteada en este trabajo esta en la función de aptitud asociada a cada modelo, dado que abordamos un problema de clasificación. De esta forma consideraremos la ecuación (7) de $l(\boldsymbol{\theta})$ como la función de error de un individuo f de la población. Por tanto definimos la medida de aptitud de un individuo mediante una función estrictamente decreciente de la función de error $l(\boldsymbol{\theta})$ en la forma $A(f)=\dfrac{1}{1+l(\boldsymbol{\theta})}$, donde $0<A(f)\le 1$.

La idea básica del algoritmo es la utilización de operadores de selección, replicación y mutación (paramétrica y estructural) en el proceso de evolución. El algoritmo se estructura en los siguientes pasos:

1.- Generar la población

2.- Repetir hasta que se cumpla la condición de parada

- Calcular la aptitud para cada individuo
- Ordenar de mayor a menor según la aptitud
- Copiar el mejor en la siguiente generación
- Replicar el r % mejor de la población por el r % peor
- Aplicar mutación paramétrica al r % mejor
- Aplicar mutación estructural al (100-r)% restante

Etapa 2. Una vez que hemos encontrado las m mejores funciones de base asociadas al individuo con una aptitud mejor en la última generación; consideramos una transformación del espacio de covariables, añadiéndoles las transformaciones no lineales del espacio de entrada, esto es, las funciones de base obtenidas mediante el algoritmo evolutivo en la etapa 1:

$$H: R^k \rightarrow R^{k+m}$$
$$(x_1, x_2, ..., x_k) \rightarrow (x_1, x_2, ..., x_k, z_1, ..., z_m)$$

donde $z_1 = B_1(\mathbf{x}, \hat{\mathbf{w}}_1), ..., z_m = B_m(\mathbf{x}, \hat{\mathbf{w}}_m)$.

Etapa 3. Aplicamos el modelo de regresión logística estándar a las variables $x_1, x_2, ..., x_k, z_1, ..., z_m$ en el nuevo espacio de características de entrada. De esta forma calculamos el máximo de la función de verosimilitud condicional para n_T observaciones:

$$l(\hat{\boldsymbol{\theta}}^1) = \sum_{i=1}^{n_T} \left\{ y_i f(\mathbf{x}, \hat{\boldsymbol{\theta}}^1) - \log(1+e^{f(\mathbf{x},\hat{\boldsymbol{\theta}}^1)}) \right\} \qquad (8)$$

donde $\hat{\boldsymbol{\theta}}^1 = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \hat{\mathbf{W}})$. Para obtener este procedimiento de estimación utilizamos un método basado en el gradiente, obteniéndose los estimadores $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{W}})$ de los parámetros del modelo:

$$f(\mathbf{x}, \hat{\boldsymbol{\theta}}) = \hat{\alpha}_0 + \sum_{i=1}^{k} \hat{\alpha}_i x_i + \sum_{j=1}^{m} \hat{\beta}_j \prod_{i=1}^{k} x_i^{\hat{w}_{ji}} \qquad (9)$$

Etapa 4.

Para obtener el modelo final, utilizaremos un método de eliminación de variables no significativas del modelo basado en un procedimiento de eliminación de una variable en cada paso, empezando con el modelo completo y eliminando las variables secuencialmente hasta que la eliminación de una variable no mejore la capacidad de clasificación del modelo sobre el conjunto de generalización. En cada etapa, eliminamos la variable menos significativa. Si utilizamos un test condicional del cociente de verosimilitudes para seleccionar uno entre dos modelos anidados, la variable eliminada en cada etapa tendrá el mayor valor del nivel crítico, cuando contrastamos la hipótesis de que su parámetro asociado toma el valor 0 en el modelo que resulta de eliminar esa variable en la selección efectuada en la etapa anterior. En nuestro caso, la primera etapa de este procedimiento consiste en efectuar contrastes de hipótesis mediante test condicionales del cociente de verosimilitudes, cada uno de los cuales contrasta el modelo completo con todas las variables frente al modelo obtenido eliminando cada una de las variables del modelo. El procedimiento termina cuando los contrastes en una etapa tienen niveles críticos inferiores a un valor fijado y el modelo seleccionado en la etapa anterior mantiene buenos porcentajes de clasificación sobre el conjunto de test.

IV. EXPERIMENTACIÓN

Los experimentos han sido planteados con el objetivo de contrastar los resultados de nuestro modelo con otros métodos de clasificación evaluando el rendimiento sobre cuatro bases de datos de prueba que clasifican los patrones en dos clases y que han sido tomadas del repositorio de la UCI [14]. El diseño experimental para los problemas propuestos fue realizado utilizando un procedimiento de validación cruzada del tipo 10-fold. Los parámetros utilizados en el algoritmo evolutivo de la etapa primera para el aprendizaje de los modelos PUNN son comunes para los 8 problemas abordados en este trabajo: Los exponentes w_{ji} se inicializan en el intervalo $[-5,5]$, los coeficientes β_j se inicializan en el intervalo $[-10,10]$. El máximo número de nodos en la capa oculta de los modelos PUNN es $m=6$; el tamaño de la población es $N_R = 1000$. El número de nodos que pueden ser añadidos o eliminados mediante una mutación

estructural está entre 1 y 2; mientras que el número de conexiones que se pueden añadir o eliminar en una mutación estructural es un número entero del intervalo [1, 3k]. Las condiciones propuestas para el criterio de parada son: o bien que en 30 generaciones no se mejore ni el rendimiento medio del 20% de los mejores individuos de la población, ni la aptitud del mejor individuo, o bien que el algoritmo alcance 100 generaciones. Los experimentos realizados muestran que Un número de generaciones mayor produce sobreentrenamiento.

V. RESULTADOS

En la tabla I describimos brevemente las cuatro bases de datos utilizadas que clasifican a los patrones en dos clases, con un número diferente de patrones, que van desde 351 hasta 1000, diferentes tipos de variables de entrada, en escala nominal, binaria o dicotómica y continua.

Para validar el rendimiento del modelo utilizamos la proporción de correcta clasificación (CCR) para el conjunto de generalización y que se define como el porcentaje de patrones de los datos correctamente clasificado. Para poder seleccionar las variables más significativas del modelo de regresión logística, utilizamos un método de eliminación de variables por pasos, utilizando el software SPSS 13.0 [24].

Con el objetivo de valorar el rendimiento de nuestro modelo LRLPU frente a otros esquemas de aprendizaje en clasificación, haremos una evaluación empírica comparándolo con los resultados más recientes [25] sobre seis de las siete bases de datos de prueba utilizadas en este trabajo (ver tabla II). En el, los autores comparan su modelo de árbol de decisión logístico, LMT, con la regresión logística (con selección de atributos, SLogistic, o con todas las covariables del modelo, MLogistic); árboles de inducción (C4.5 [26]); y remuestreo para árboles de decisión usando como clasificador base el algoritmo C4.5 y el algoritmo AdaBoost.M1 con 100 iteraciones de remuestreo.

Para estas cuatro bases de datos, en una de ellas los mejores resultados se obtienen con AdaBoost (100), mientras que en las otras tres los mejores resultados se obtienen con SLogistic, MLogistic. En las bases de datos Australian Card y German, los resultados del modelo LRLPU están por encima de la media del mejor de los cinco algoritmos comparados.

Por otra parte, queremos destacar que los modelos LRLPU producen clasificaciones más precisas (en cuanto a la proporción de patrones bien clasificados para el conjunto de generalización) que los modelos de regresión logística con las covariables iniciales (SLogistic y MLogistic).

VI. CONCLUSIONES Y TRABAJOS FUTUROS

En este trabajo, nos hemos centrado en los problemas de clasificación binaria, para los que proponemos métodos de regresión logística basados en la combinación de un modelo lineal y un modelo no lineal formado por modelos de redes neuronales de unidades producto. Las funciones base del modelo no lineal permiten reconocer las posibles interacciones de alto grado existentes entre las covariables del modelo, a través de la multiplicación de transformaciones potenciales de dichas variables.

La metodología para el aprendizaje de los coeficientes del modelo se basa en la combinación por etapas de un algoritmo evolutivo que nos determine la estructura básica del modelo no lineal de unidades producto y de un procedimiento de optimización local basado en maximizar el logaritmo de la función de verosimilitud que nos sirve para estimar los coeficientes de regresión del modelo final. Además utilizamos un método de selección de características para seleccionar aquellas covariables que mejor expliquen la variable de respuesta. De esta forma, controlamos el número de coeficientes del modelo final y disminuimos el riesgo de construir modelos complejos que sobreaprendan los datos del conjunto de entrenamiento.

El modelo propuesto lo hemos aplicado a cuatro bases de datos de prueba mejorando en todos los casos los resultados obtenidos con el modelo de regresión logística utilizando todas las covariables iniciales. De esta forma, el modelo híbrido determina un buen balance entre considerar sólo un modelo lineal o sólo un modelo no lineal. Además, los resultados muestran que nuestra metodología es muy prometedora en términos de precisión en la clasificación. Por último, hay que destacar que nuestros resultados son competitivos con los resultados que habitualmente se citan para estas bases de datos.

AGRADECIMIENTOS

Este trabajo ha sido financiado en parte por el proyecto TIN2005-08386-C05-02 de la Comisión Interministerial de Ciencia y Tecnología y fondos FEDER.

REFERENCIAS

[1] Vapnik, "The nature of Statistical Learning Theory," Springer, 1999.

[2] Y. Freund and R. Shapire, "Experiments with a new boosting algorithm," presented at Machine Learning: Proceedings of the Thirteenth International Conference, San Francisco, 1996.

[3] P. McCullagh and J. A. Nelder, Generalized Linear Models, 2nd edn. London, 1989.

[4] R. L. Prentice and R. Pike, "Logistic disease incidence models and case-control studies," Biometrika, vol. 66, pp. 403-411, 1979.

[5] W. Vach, R. Robner, and M. Schumacher, "Neural Networks and logistic regression: Part II,"

Computational Statistics & Data Analysis, vol. 21, pp. 683-701, 1996.
[6] M. Schumacher, R. Robner, and W. Vach, "Neural networks and logistic regression: Part I," Computational Statistics & Data Analysis, vol. 21, pp. 661-682, 1996.
[7] T. Hastie, R. J. Tibshirani, and J. Friedman, "The Elements of Statistical Learning. Data mining, Inference and Prediction," in Springer, 2001.
[8] M. Bishop, Neural Networks for Pattern Recognition: Oxford University Press, 1995.
[9] J. Friedman and W. Stuetzle, "Proyection pursuit regression," Journal of the American Statistical Association, vol. 76, pp. 817-823, 1981.
[10] T. J. Hastie and R. J. Tibshirani, Generalized Additive Models. London: Chapman & Hall, 1990.
[11] J. Friedman, "Multivariate adaptive regression splines (with discussion)," Ann. Stat., vol. 19, pp. 1-141, 1991.
[12] K.-A. Tohn, "Training a reciprocal-sigmoidal classifier by feature scaling-space," Machine Learning, vol. Published online, 2006.
[13] R. Durbin and D. Rumelhart, "Products Units: A computationally powerful and biologically plausible extension to backpropagation networks," Neural Computation, vol. 1, pp. 133-142, 1989.
[14] C. Blake and C. J. Merz, " UCI repository of machine learning data bases," www.ics.uci.edu/ mlearn/MLRepository.thml, 1998.
[15] D. W. Hosmer and S. Lemeshow, Applied logistic regression. New York: John Willey & Sons, 1989.
[16] T. P. Ryan, Modern Regression Methods. New York: Wiley, 1997.
[17] G. McLachlan, Discriminant analysis and statistical pattern recognition. New York: Yohn Wiley & Sons, 1992.
[18] L. R. Leerink, C. L. Giles, B. G. Horne, et al., "Learning with products units," Advances in Neural Networks Processing Systems, vol. 7, pp. 537-544, 1995.
[19] M. Schmitt, "On the Complexity of Computing and Learning with Multiplicative Neural Networks," Neural Computation, vol. 14, pp. 241-301, 2001.
[20] A. Ismail and A. P. Engelbrecht, "Training products units in feedforward neural networks using particle swarm optimisation.," presented at Development and practice of Artificial Intelligence Techniques, Proceeding of the International Conference on Artificial Intelligence, Durban, South Africa, 1999.
[21] E. A. P. Ismail A., "Global optimization algorithms for training product units neural networks," presented at International joint conference on neural networks ijcnn`2000, 2000.
[22] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," IEEE Transactions on Neural Networks, vol. 5 (1), pp. 54-65, 1994.
[23] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," IEEE Transactions on Neural Networks, vol. 8 (3), pp. 694-713, 1997.
[24] I. SPSS ©, "SPSS ©9.0 advanced models," Inc, S., Ed. Chicago, IL, 1999.
[25] N. Landwehr, M. Hall, and F. Eibe, "Logistic Model Trees," Machine Learning, vol. 59, pp. 161-205, 2005.
[26] R. Quinlan, C4.5: Programs for Machine Learning: Morgan Kauffman, 1993.

Tabla 1. Descripción de las bases de datos utilizadas

Base de Datos	Casos	Variables C	B	N	T	Descripción
Breast-Cancer	699	9	0	0	9	Existen dos clases de forma tal que se clasifican a los enfermos con cáncer, como benigno (el 65.5%) o maligno (el 34.5%).
Ionosphere	351	33	1	0	34	Señales “buenas” del radar son aquellas que muestran la evidencia de que hay algún tipo de estructura en la ionosfera y señales “malas” son aquellas en las que no la hay.
Australian-Card	690	6	4	5	51	Existen dos clases, indicando si la solicitud de tarjeta de crédito fue aceptada (el 44.5%) o denegada (el 55.5%).
German	1000	6	3	11	61	Los atributos que están en escala binaria o nominal se han codificado como enteros, como se hace en el proyecto StatLog.

Tabla 2. Resultados estadísticos del algoritmo LRLPU y de los métodos Logistic Model Tree (LMT), SLogistic, MLogistic, C4.5, ABoost(100).

Dataset	LMT	SLogistic	MLogistic	C4.5	ABoost(100)	LRLPU
Breast-cancer	74,91±6,29	**74,94±6,25**	67,70±6,92	74,28±6,05	66,36±8,18	71,40±4,47
Ionosphere	92,99±4,13	87,78±4,99	87,72±5,57	89,74±4,38	**94,02±3,83**	91,99±5,67
Australian Card	85,04±3,84	85,04±3,97	85,33±3,85	85,57±3,96	86,43±3,98	**87,55±3,89**
German	75,37±3,53	75,34±3,50	75,24±3,54	71,25±3,17	74,53±3,26	**76,90±5,20**