UNIVERSIDAD DE CHILE
FACULTAD DE ECONOMIA Y NEGOCIOS
ESCUELA DE ECONOMIA Y ADMINISTRACION

Aproximación al Desarrollo de Procesos Automatizados de Selección de Funciones de Activación en Redes Neuronales, y Evidencia de sus Efectos

Seminario para optar al Título de Ingeniero Comercial
con Mención en Administración

Rubén Humberto Catalán Cabezas

Profesor Guía: Pablo Tapia Griñén

Diciembre 2008

Abstract

Esta investigación evidencia las ventajas de automatizar el funcionamiento de una red neuronal, en el contetxto de aplicaciones financieras de valoración de activos y portfolios. Demostramos que el establecimiento de forma manual de las funciones de activación tiene efectos adversos sobre la efectividad de la red; en términos de los niveles de error alcanzados. Se encuentra que el hacer una elección óptima puede disminuir entre un 20% y un 95% el ECM del modelo. Además se desarrollan las bases de un modelo teórico de resolución de Redes Neuronales que no requiere de la selección de una función de activación, ya que éstas son resultado del proceso de optimización del modelo.

Dedicatoria

Dedico esta tesis especialmente a mi padre y a mi madre, por mostrarme que
la familia es el mejor motivador para el esfuerzo, y que
el esfuerzo es uno de los mejores regalos para una familia.

Se la dedico además a María Paz, mi polola, porque
gracias al amor y apoyo brindado, poco a poco se
está conviertiendo en la motivación de mis esfuerzos.

Agradecimientos

Agradezco profundamente a Pablo Tapia, quién mi guió en esta tesis, no sólo por haber ayudado a mejorar esta tesis mucho más de lo que mis capacidades hubieran permitido, sino que también por los valiosos consejos, y las conversaciones y discusiones mantenidas durante mi estadía en la Facultad.

Asimismo debo expresar mis agradecimientos a Félix Lizama, profesor del cual fui ayudante por más de dos años, porque me ha permitido desarrollar otras competencias y habilidades usualmente escasas en las aulas universitarias.

Tabla de contenido

Introducción

En esta tesis se evidencian las ventajas de automatizar el funcionamiento de una red neuronal, en el contetxto de aplicaciones financieras de valoración de activos y portfolios. Se postula que el establecimiento de forma manual de ciertos parámetros de la red, como la cantidad y organización de neuronas, o la selección de funciones de activación tiene efectos adversos sobre la efectividad de la red; ya sea en términos de la capacidad de la red para abordar el problema, como en los niveles de error alcanzados. Enfocándonos en la selección de las funciones de activación mostramos que la elección no es irrelevante en términos de error ni de capacidad predictiva, proponiendo un método de optimización que elimina la necesidad de seleccionar manualmente una función de activación.

La literatura sobre Redes Neuronales (en adelante NNs – Neural Networks) e Inteligencia Artificial es amplia, desde el desarrollo de la neurona de McCulloch & Pitts, y del Perceptron por Rosenblatt a finales de la década del 50[1], pasando por una época de boom en los años 80, hasta la actualidad en la que el desarrollo teórico de métodos y modelos sigue avanzando. Prueba de ello es la c de Journals que tratan directa o tangencialmente el campo: *Journal of Neural Networks, Journal of Neural Computing & Applications, Journal of Artificial Intelligence Research,* entre otros. Por otra parte estas técnicas ya se han convertido en herramientas comprobadamente útiles para realizar diversos tipos de análisis, y se encuentran integradas en sistemas de alta sofisticación técnica. Por ejemplo el sistema electrónico del Aston Martin DB9 utiliza NNs para prevenir fallos en el proceso de ignición del motor (Block 2004), o mediante el desarrollo del *Intelligent Flight Control System*, en el Dryden Flight Research Center de la NASA, las NNs son capaces de controlar aviones F-15 en vuelo, y corregir en tiempo real parámetros de vuelo del piloto automático ante cambios en las condiciones atmosféricas externas (Dryden Flight Research Center, NASA 2006). Variados ejemplos de otras aplicaciones de NNs son explicados en (Peltarion s.f.)

[1] Ver McCulloch and Pitts (1943) y Rosenblatt (1958).

Las redes neuronales también han tomado protagonismo en el manejo de aplicaciones financieras, sobre temas tales como, solo por citar algunos ejemplos, inversión en divisas (Colin 1992) , manejo de portfolios (Leigh, Purvis y M. 2002), y medición de riesgos (Tsai y Wu 2008), entre otros. Estos son asuntos claramente vitales dentro del manejo financiero de una empresa, especialmente si hablamos de clasificadoras de riesgo, bancos o instituciones de seguros, cuyo giro es específicamente administración financiera de activos.

En este contexto de aplicación de las redes neuronales es de interés en esta tesis apuntar hacia un tema que la literatura ha dejado de lado, y que potencialmente puede tener fuertes efectos dentro del desempeño de estos modelos: la elección por parte del investigador en base a criterios no probados, y muchas veces errados, de ciertos aspectos en el modelamiento de las NNs. Un ejemplo de este tipo de elecciones, en el que se enfoca esta tesis, son las funciones de activación.

De un modo resumido, esta elección ocurre en el siguiente contexto: los datos utilizados para optimizar el modelo de NNs son tratados y modificados en distintas formas durante el proceso de estimación. En una de estas etapas de modificación es donde se encuentran involucradas las funciones de activación. Esta función participa en el cálculo del resultado de cada neurona de la red, afectando el proceso de estimación, y más importante, la calidad de la estimación. Esto es debido a que, como demostraremos durante el desarrollo de esta tesis, los resultados del proceso son efectivamente diferentes dependiendo de la elección de función hecha por el investigador.

Justamente la elección de esta función es el tema que atañe a esta tesis. En variadas investigaciones se utilizan configuraciones especiales de la red, que difieren en su efectividad de tratar diferentes problemas: clasificación, predicción, estimación funcional, etc. Estas configuraciones normalmente especifican cuántas capas hay, cuántas neuronas tienen las capas ocultas, se definen las interrelaciones entre las capas, y cuáles son las funciones de activación utilizadas. Ahora bien, las razones que guían tal decisión normalmente no son explicadas, ni tampoco se evalúan los efectos de decidirse por una u otra. Por mencionar dos ejemplos, tanto González y Jiménez (2003) como Gutiérrez (2004) podrían estar cometiendo

errores de especificación del modelo al abordar el problema de la selección de funciones de activación. Ello es debido a que, dentro de la gama de funciones posibles, simplemente toman y aplican algunas, sin evaluar los efectos o costos de la utilización de alguna de ellas.

Tenemos como objetivos secundarios de esta tesis el establecer las bases de un modelo teórico que permita al proceso, por sí mismo, el tomar la decisión de seleccionar la función que permita obtener mejores resultados, como parte del proceso de optimización de la red, y demostrar que efectivamente la elección de esta función tiene efectos sobre los resultados del modelamiento.

Respecto al primero, el diseñar un método que en base a criterios objetivos como la minimización del error cuadrático medio del modelo, tome esta decisión automáticamente y sin intervención humana, presenta un alto beneficio respecto de eliminar o al menos disminuir los sesgos introducidos en el modelo por el investigador al escoger, en base a criterios poco claros, estos parámetros de configuración de la red, como sucede en los estudios antes citados. No obstante, lo que se pretende es sólo sentar las bases de dicho modelamiento: definir los criterios generales, y discutir las alternativas de modelamiento, y no el de desarrollar completamente un modelo de dichas características. Por otra parte, el segundo objetivo implica dos cosas: demostraría que se pueden establecer métodos de verificación de si una elección es óptima, es decir que criterios teóricos que definen la mejor capacidad de un modelo pueden ser verificados empíricamente a costos razonables en términos de procesamiento. Lo segundo, y más importante, es que validaría la hipótesis central de esta tesis de que el investigador está incurriendo en un riesgo importante al escoger la función de activación sin tomar en cuenta los efectos que esto pudiera tener en los resultados del modelo final.

En la práctica, lograr ambos objetivos, y posteriormente el ser capaz de incluir en el modelamiento dichos aspectos significará poder tomar mejores decisiones con NNs: por ejemplo, disminuir los errores en los que cae el modelo al predecir los impagos en una institución financiera, tener un menor error de valoración de activos, o diseñar modelos de portfolios accionarios que manejen de mejor forma el riesgo y las volatilidades de los mercados.

Este trabajo se distribuye de la siguiente manera: Sección I. Aspectos Teóricos, se muestran los aspectos tradicionales de la teoría de redes neuronales, el diseño y funcionamiento de una red, y el desarrollo del algoritmo de optimización tradicional, el algoritmo de *backpropagation.* Posteriormente se discuten las funciones de activación más comunes, la manera en que tradicionalmente se escogen en las aplicaciones de modelos, y además cómo se relacionan con el algoritmo de optimización. En el último punto mostraremos una teoría de formación de funciones originalmente postulada por Beresteanu (2007). Sección II. Bases de un Modelo Integrado de Optimización, intenta resolver nuestro primer objetivo, el de comenzar a desarrollar un modelo que aborde los problemas de elección funcional. Sección III. Evidencia de efectos de la elección funcional, se demuestra el efecto empírico de una mala elección funcional. La Sección IV. Conclusiones resume los mayores aportes y finaliza la investigación.

Sección I. Aspectos Teóricos

Redes Neuronales

Las redes neuronales son capaces de establecer relaciones no necesariamente lineales entre los inputs y una variable objetivo. La idea es entrenar a la red para lograr que reconozca esta relación, y entonces realizando cálculos en base a los inputs, aproxime los resultados de la variable objetivo.

Para comprender este proceso, revisaremos primero el funcionamiento de su componente más básica: una neurona. Éstas son unidades de cálculo que generan un resultado, partiendo de variados tipos de inputs. Haciendo el paralelo con la neurona biológica, los inputs pueden considerarse como estímulos: la neurona biológica responde de diferentes maneras, dependiendo de la fuerza del estímulo inicial; la neurona artificial en tanto, toma el estímulo -que en nuestro caso corresponde a datos numéricos-, y los transforma mediante operaciones aritméticas o funciones más elaboradas. El resultado de este proceso es el output de la neurona. Posteriormente este output puede seguir siendo procesado por la red, considerándolo como si fuera un input. Ello hace que a grandes rasgos una red neuronal sea un set de neuronas interconectadas entre sí, alimentándose unas a otras de información. Las series de variables y datos iniciales sirven como input para las neuronas de la capa inicial, en donde comienzan a ser modificados y procesados por la red, hasta que el resultado de la neurona final corresponde a la predicción que hace la red.

El siguiente esquema[2] muestra el funcionamiento de una neurona:

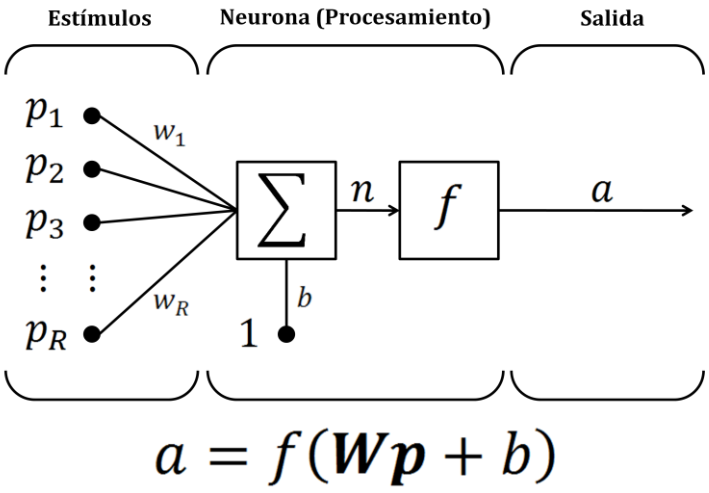


Figura 1 - Esquema de Funcionamiento Neuronal, Basado en Hagan, Demuth y Beale (1996)

Los R estímulos o inputs p_i son multiplicados cada uno por un w_i y luego sumados. Este proceso de ponderación es representado en la forma funcional inferior por $\boldsymbol{Wp}$**.** Opcionalmente puede sumarse un valor b, constante al factor calculado. Luego, el escalar resultante, llamado n, es procesado por medio de una función de activación f. Esta función de activación puede tomar diversas formas, que serán estudiadas con detalle más adelante. El resultado de la función es el output de la neurona, llamado a. Todo este proceso es representado matemáticamente por,

$$a = f(\boldsymbol{W}\boldsymbol{p} + b) \quad (1)$$

De la misma manera como se calculó a, el resultado de una neurona, es posible calcular el resultado de S neuronas diferentes, dado a que dispongamos de S sets de ponderadores

[2] Los esquemas mostrados son modificaciones de aquellos desarrollados por Hagan, Demuth y Beale (1996).

diferentes, y S diferentes valores para b. Cuando estas S neuronas se basan en los mismos R inputs, y utilizan la misma función de activación, este conjunto de neuronas es conocido como una capa neuronal.

Esquemáticamente el diagrama de una capa neuronal es:

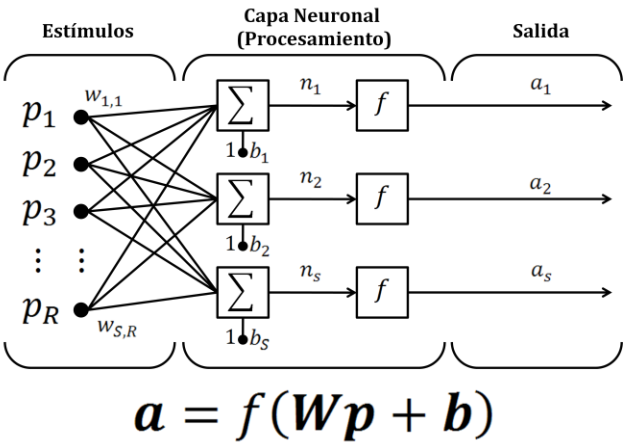


Figura 2 - Esquema de Funcionamiento de una Capa, Basado en Hagan, Demuth y Beale (1996)

Que podemos esquematizar con una notación abreviada:

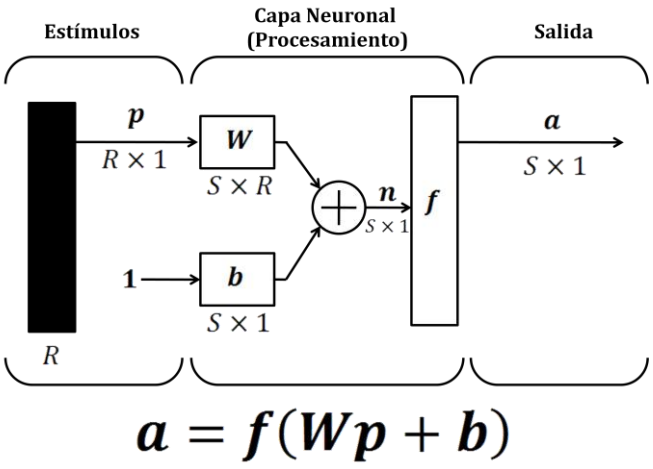


Figura 3 - Esquema Abreviado de Funcionamiento de una Capa, Basado en Hagan, Demuth y Beale (1996)

Es importante notar que el vector $\boldsymbol{R}$, representa R inputs para esta neurona. Al comenzar el proceso de estimación, estos valores son ponderados por una matriz $\boldsymbol{W}$, de S filas y R columnas, donde S es la cantidad de neuronas en la capa. Vale mostrar nuevamente que no necesariamente la cantidad de inputs es igual a la cantidad de neuronas en una capa. El producto es sumado a una matriz $\boldsymbol{b}$, opcionalmente[3]. En este punto del proceso tenemos un vector fila de S elementos que corresponden a valores asociados a cada una de las S neuronas. Finalmente el resultado corresponde la evaluación en $\boldsymbol{f}$ del vector S.

En términos matemáticos, este resultado es posteriormente utilizado como input de la siguiente neurona, tendremos que una red de z capas equivale a la composición de z funciones. Por ejemplo, para una red de 3 capas:

$$\hat{y} = a^3 = f^3(LW^{3,2}f^2[LW^{2,1}f^1\{IW^{1,1}p + b^1\} + b^2] + b^3) \quad (2)$$

La estimación resultante del modelo $\hat{y}$, corresponde al output de la tercera neurona, a^3, y es igual a una composición de los procesos de 3 neuronas. El supraindice de los factores indica la capa a la que pertenecen. Por ejemplo, $LW^{3,2}$ es la matriz de pesos que conecta las capas 2 y 3, y f^3 la función de activación del modelo en la capa 3.

Estimación

El proceso de estimación de la red no es único. En la literatura se han desarrollado distintos métodos de optimización que apuntan a resolver variados problemas de la formulación original del modelo: la trampa de caer en mínimos locales, enfoques numéricos para resolver problemas de procesamiento computacional, la eficiencia de la optimización del modelo, entre otros, los que difieren fundamentalmente en el algoritmo establecido para la obtención del mínimo.

[3] En la literatura el valor de $\boldsymbol{b}$ se conoce como 'sesgo', lo que contradice nuestra intuición ya que hace pensar que se le agregará un sesgo al resultado de la neurona. Esto es exactamente lo contrario, ya que la intuición es similar a la del caso de una regresión OLS: el incluir el término $\boldsymbol{b}$ le permite al resultado del modelo precisamente no estar obligado a pasar por 0 para un vector $\boldsymbol{R} = \boldsymbol{0}$.

Para el desarrollo completo del algoritmo de *backpropagation*, ver Demuth, Beale y Hagan (2008). Además este libro provee información de otras técnicas de optimización y diseño de NNs. En el estudio de Mutihac y Hulle (2003) se muestra una recopilación de algoritmos de optimización de NNs para análisis de componentes independientes; Brent (1973) publica su texto *Algorithms for Minimization without Derivatives*, en donde se elaboran técnicas de búsqueda lineal, y en Demuth, Beale y Hagan (2008) encontramos una referencia a variadas técnicas de NNs preprogramadas en Matlab. En la literatura ya se han desarrollado variados métodos de optimización, que tienen efectos respecto a la rapidez con la que se optimiza la red, y en la capacidad de la red de predecir correctamente los valores objetivo. Métodos como G*radient Descent, Gradient Descent with Momentum,* o *Resilient Backpropagation* dependen directamente de las derivadas de las funciones de activación, mientras que métodos como *Conjugate Gradient, Quasi-Newton Algorithms,* o *Levenberg-Marquardt Algorithms* buscan reemplazar la matriz de Hess del proceso de optimización utilizando combinaciones de aproximaciones a la primera derivada calculadas numéricamente.

Para objeto de esta tesis desarrollaremos nuestras mejoras sobre una implementación acotada, que no pretende abordar todos los problemas descritos anteriormente, sólo aquellos que son tratados en el desarrollo teórico. Para la implementación de la programación de los métodos nos basaremos en un algoritmo primitivo, conocido como *algoritmo de propagación hacia atrás (backpropagation)*, y es el que nos servirá para sustentar el desarrollo de las bases del modelo teórico del siguiente apartado. A continuación se mostrará el método de *backpropagation*, y la implicancia que tiene el hacer cambios en la función de activación del modelo.

Algoritmo

Para verificar la calidad de un modelo de redes neuronales, y en general cualquier proceso de estimación estadístico, se utilizan variadas medidas de eficiencia sobre las cuales es posible evaluar los resultados: el Error Medio Absoluto, Error Cuadrático Medio, Suma de los Errores Cuadráticos, etc. El algoritmo de *backpropagation* es desarrollado a partir de la optimización sobre un criterio de error cuadrático medio (Ψ), descrito como:

$$\Psi(\Omega) = \frac{1}{n} E[(\boldsymbol{t} - \boldsymbol{a})'(\boldsymbol{t} - \boldsymbol{a})] = \frac{1}{n} \boldsymbol{e}'(k)\boldsymbol{e}(k) \quad (3)$$

Los factores que el proceso debe ajustar para minimizar el ECM son el conjunto de ponderadores y sesgo:

$$\Omega_{\mathrm{j}}: \left\{ IW_j^{c_t, c_i}, b_j^{c_k} \right\} \ \forall\ k, i, t\ \in \boldsymbol{M} \quad (4)$$

Es decir, aquellos factores escalares pertenecientes a las matrices de ponderadores que intervienen entre los outputs de la capa anterior de neuronas (la capa c_i, o capa de inicio), y la neurona actual (c_t, o capa de término), para todas aquellas combinaciones pertenecientes a $\boldsymbol{M}$, el conjunto de capas de la red. En nuestro modelo esquemático (ver figura 3), los factores a actualizar corresponden a la matriz $\boldsymbol{W}$, y a la matriz $\boldsymbol{b}$.

El modelo se resuelve por medio de *backpropagation* al ir calculando en distintas iteraciones, valores de Ω_{j} que mejor ajusten el modelo (es decir, que entreguen menor ECM). En cada iteración j del modelo se obtiene un nuevo set de parámetros Ω_{j}. El proceso iterativo se inicia con un set de ponderadores dados Ω_0, obtenidos a través de un proceso de generación de números aleatorios en el intervalo $[0{,}1]$.

Como se mencionó, el proceso de optimización establece la minimización del ECM respecto del conjunto de ponderadores Ω. Como se ve claramente en la ecuación (2), el resultado de la red a^M, y por lo tanto el ECM, dependen implícitamente de la elección de la función de traspaso f. Por ello al derivar para encontrar los óptimos se utiliza extensamente la regla de la cadena. La razón de esto es que la función de traspaso f está presente tantas veces como capas haya en la red.

El resultado de esta optimización puede esquematizarse como se muestra en la ecuación (6), sujeto a que dentro del proceso de optimización de $\Psi(\Omega)$ se defina un parámetro de sensibilidad del error[4]:

$$s_i^m \equiv \frac{\partial \hat{\Psi}}{\partial n_i^m} \tag{5}$$

Este parámetro corresponde al grado de variación en la función ECM respecto del cambio en el resultado de la neurona m.

Al definir este parámetro es posible reordenar el resultado de nuestro proceso de optimización para establecer el algoritmo de actualización de los datos como:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha s_i^m a_j^{m-1}$$
$$b_i^m(k+1) = b_i^m(k) - \alpha s_i^m \tag{6}$$

Es decir, se toma el valor del parámetro (ya sea tanto w como b) en la iteración k, se ajusta por medio del parámetro de sensibilidad y los resultados de las neuronas, para obtener el valor del parámetro en la siguiente iteración. Estas son condiciones de recurrencia en base a las condiciones de primer orden de la optimización.

El resultado de estos parámetros, que corresponde a nuestro set de ponderadores Ω, nos permite establecer los pesos de la red, dejándola lista para efectuar predicciones.

[4] Para un desarrollo paso a paso del método de Backpropagation, puede revisarse el capítulo 11 de (Hagan, Demuth y Beale 1996).

Funciones de Activación

Funciones Tradicionales

Como quedó de manifiesto en el punto anterior, las funciones de activación afectan el parámetro de sensibilidad definido en (5), y por lo tanto tienen efectos en el proceso de actualización de pesos que define el óptimo (ecuación 6). Por ello dedicaremos esta sección a revisar los tipos de funciones usadas.

Una función de activación es una función que responde a los valores del escalar n, y que no necesariamente es lineal. Encontramos las siguientes familias funcionales[5]:

1. Dicotómicas. Funciones que generan sólo dos posibles outputs.
2. Lineales. Funciones que devuelven la representación lineal del input.
3. Logísticas. Funciones que devuelven cambios graduales entre dos niveles.
4. Otras. Funciones que integran otro tipo de comportamientos.

En la literatura no es frecuente encontrar información respecto a qué tipo de funciones tienen mayor efectividad en distintos tipos de redes, a excepción del comentario de que la función *Hard Limit* básica es usada por el Perceptron para resolver problemas de clasificación binaria (McNelis 2005).

En términos de eficiencia es común observar comparaciones entre distintos algoritmos de optimización, por ejemplo en Chen, Racine y Swanson (2001). Estos análisis tienen la particularidad de reflejar los costos asociados a métodos de optimización diferentes, lo que en ocasiones entrega soluciones diferentes. Por ejemplo algoritmos que caen en trampas de mínimos locales, o inclusive soluciones iguales, pero con un tiempo de optimización diferente.

[5] Ver formas funcionales en Anexo "- Esquematización de Funciones Comunes", página 36.

Esta tesis se desarrollará de forma bastante similar a aquellos estudios, pero con la diferencia de que tiene como hipótesis encontrar que efectivamente existen diferencias para modelos realizados con los mismos algoritmos, los mismos datos, e inclusive la misma configuración en término de las capas neuronales, al cambiar sólo la elección de la función de activación.

Otro tipo de análisis común de encontrar en la literatura es el de comparativas de configuraciones de redes que resuelven cierto tipo de problemas, como en el caso del Perceptron comentado, y por lo tanto que especifican qué función se debe utilizar, y cuál es la cantidad de neuronas y capas que debe tener la red. Ello no invalida el objetivo de la tesis, ya que el problema de fondo sigue manteniéndose presente: no se hace una revisión de los efectos que la elección de la función tiene en el resultado, sesgo y eficiencia del proceso de optimización.

Proceso Tradicional de elección de las funciones a utilizar en el Modelamiento

Como se explicó anteriormente, en la literatura aplicada el investigador utiliza configuraciones especiales de la red neuronal, que difieren en su efectividad de tratar diferentes problemas: clasificación, predicción, estimación funcional, etc. Estas configuraciones normalmente especifican cuántas capas hay, cuántas neuronas tienen las capas ocultas, se definen las interrelaciones entre las capas, y cuáles son las funciones de activación utilizadas. Ahora bien, las razones que guían tal decisión normalmente no son explicadas, ni tampoco se evalúan los efectos de decidirse por una u otra.

En particular respecto a la elección de funciones, abundan ejemplos: En González y Jiménez (2003), se explica que utilizan una función de activación Gaussiana y una Tangente, pero a ni siquiera se intenta explicar la razón de su uso, o elaborar los posibles efectos de un cambio. Gutiérrez (2004) utiliza en su modelo dos funciones: una función Gaussiana, y una Gaussiana complemento, y dedica un párrafo para entregar razones que soporten esta elección: *"(la utilización de estas funciones) permite que la red tenga dos puntos de vista de la misma información, con lo que se puede lograr un resultado más adecuado a la realidad"*, con lo que

deja por cerrado el punto. Claramente esta explicación es insuficiente, al no evaluar alternativas, ni especificar algún criterio que respalde su afirmación.

Desarrollo Funcional

Arie Beresteanu desarrolla modelos no paramétricos con un método de mínimos cuadrados en el cual el proceso de optimización por si solo determina cuál es la función de regresión objetivo proporcionando sólo restricciones sobre sus derivadas parciales –(ver Beresteanu 2007). Si bien el método desarrollado está ligado estrechamente a la estimación de este tipo de modelo noparamétrico, su aplicación en el campo de las redes neuronales puede efectivamente ser la solución al problema planteado anteriormente.

El método de Beresteanu comienza desarrollando una grilla de r puntos, que forman una función $\Phi_r(x) = \{\phi_{r,1}(x), \phi_{r,2}(x), \dots, \phi_{r,r}(x)\}$. La grilla se conforma por la unión de r otras funciones, todas definiendo el valor de Φ_r cuando $x = 1/r$. Para hacer que la función sea continua, el resto de los valores no pertenecientes a la grilla, es decir entre un x y el siguiente, son interpolados a partir de los valores de $\phi_{r,1/x}(x)$ y de $\phi_{r,1/x}(x+1)$.

La forma funcional de $\phi_{r,j}(x)$ se define como

$$\phi_{r,j}(x) = \begin{cases} 1 - |rx - j| & x \in \left[\frac{j-1}{r}, \frac{j+1}{r}\right] \cap [0,1] \\ 0 & otro\ caso \end{cases} \quad (7)$$

Lo que corresponde a una *triangular kernel function*.

El segundo paso del método de Beresteanu corresponde a la imposición de restricciones de forma, que fundamentalmente corresponde a restricciones sobre los valores que toman las derivadas de la función. Como la función a optimizar es construida sobre una grilla, su derivada no es analítica sino que también se construye sobre una grilla. La particularidad de esta grilla-derivada es que se imponen restricciones sobre las diferencias entre los puntos, representadas por la siguiente matriz:

$$A_r^p g \geq 0 \tag{8}$$

La matriz A_r^p contiene ponderadores que indican la estructura de diferencias entre los valores de la grilla, y g es un vector que toma valores dependiendo de los resultados de la función.

Finalmente, el tercer paso del método de estimación de Beresteanu corresponde a la interpolación de los valores funcionales para aquellos puntos intermedios, no presentes en la grilla. Dependiendo de la elección de la función generadora $\phi_{r,j}(x)$, este método puede ser más o menos complejo. En particular la elección de la *triangular kernel function* simplifica el proceso, tanto al establecer las restricciones de forma, como al momento de interpolar. Beresteanu discute otro tipo de caso, en donde las funciones son más generales: *bases B-Spline de mayor orden*, y que permiten imponer restricciones sobre los valores de la 3ra derivada en adelante.

Un punto a tener en cuenta es la cantidad de grados de libertad necesarios en el modelamiento: debido a que lo estimado es una grilla, el proceso es intensivo en el uso de datos. Si la base disponible no incluye gran cantidad de puntos, existe la alternativa de semiparametrizar el modelo.

Este método de composición funcional no necesariamente es nuevo en la literatura: una aplicación similar es encontrada en la ingeniería financiera. Es posible armar un esquema de pagos contingentes a un estado de la naturaleza futuro mediante cierto tipo de activos derivados. Específicamente, son bastante conocidos esquemas de pagos basados en opciones como el *straddle*, *butterfly*, y los *bearish* o *bullish spreads*. Mediante una metodología similar a la de Beresteanu, es posible aproximar cualquier función de pago contingente deseada (Hull 2006), en base a *Butterfly Spreads*, y tomando en cuenta las restricciones impuestas por los costos de los derivados financieros.

Sección II. Bases de un Modelo Integrado de Optimización

Es de interés el poder desarrollar un modelo que automáticamente, en base a algún criterio objetivo –como el de ECM, por ejemplo-, pueda ser capaz de integrar la elección óptima de una función de activación dentro del proceso de estimación de la red. En el siguiente apartado se delineará cómo estimar un modelo que tenga tales capacidades.

Premisa

Como quedó de manifiesto en la sección de optimización, podemos desarrollar una expresión general para la formulación de (2) , lo que es equivalente, en una red de k capas a:

$$a^k = f^k\left(LW^{k,k-1} f^{k-1}[LW^{k-1,k-2} f^{k-2}[\dots] + b^{k-1}] + b^k\right) = \hat{y} \tag{9}$$

$$a^k = g\left(\{LW^{i,i-1}\}_{i=1}^{k}, \{f^i\}_{i=1}^{k}, \{b^i\}_{i=1}^{k}\right) \tag{10}$$

Donde g es una función no lineal de LW, f y b.

Ahora bien, en el proceso de optimización h[6],

$$h = \min_{LW}\{\Psi\} = h(g(\dots), \Psi, p, a) \tag{11}$$

Desde el punto de vista de la optimización funcional, no numérica, podemos resolver el Lagrangeano como una regla de la cadena:

$$\frac{\partial h}{\partial LW^{i+1,i}} = \frac{\partial h}{\partial g} \cdot \frac{\partial g}{\partial f^k} \cdot \frac{\partial f^k}{\partial f^{k-1}} \cdot \frac{\partial f^{k-1}}{\partial f^{k-2}} \cdot \dots \cdot \frac{\partial f^{i+1}}{\partial f^i} \cdot LW^{i+1,i} f^i\left[f^{i-1} LW^{i,i-1} f^{i-1}(\dots)\right] \tag{12}$$

Pues bien, es efectivamente sobre el conjunto de derivadas $\partial f^{i+1}/\partial f^i$ en donde podemos aplicar tanto la metodología como las restricciones de forma que desarrolló Beresteanu (2007).

[6] En este caso, el proceso h corresponde al algoritmo de Backpropagation.

Luego, sobre este proceso de optimización es posible obtener un algoritmo que además de definir los valores del set de ponderadores Ω, establezca los valores para la función kernel, por lo tanto, definiendo $\Phi_{r.}$ como una función de activación óptima.

Restricciones

Ahora bien, para implementar dicha solución nos encontramos con dificultades prácticas. Si el set de funciones $\{f^i\}_{i=1}^{k}$ incluye funciones de diferente especificación, la cantidad de grados de libertad necesarios en la estimación puede ser prohibitiva. Esto es debido a que por la metodología de Beresteanu tenemos que, para cada función a estimar, poder determinar los parámetros que definen los valores de la función en cada punto de la grilla de datos. Por ejemplo, si definimos una grilla con una cantidad moderada de puntos: 10, y estamos hablando de una cantidad de 3 capas ocultas, con 5 neuronas cada una, tenemos un total de 150 valores a estimar. En la práctica, hacerlo para sets de datos pequeños no es posible, porque los grados de libertad disponibles en dicho tipo de sets es muy limitada. Una alternativa en ese tipo de casos es establecer que todas las capas tengan la misma función de activación, caso en el que limitamos los parámetros a estimar, ya que corresponden sólo a los de una función, y no a una por cada capa. Ello supondría que los grados de libertad usados por el modelo bajan de 150 a 10.

La segunda dificultad es relativa al objetivo del método. Esto es debido a que el problema al que se aplica el modelo de Beresteanu corresponde a definir una función que aproxime analíticamente de la mejor forma a otra función empírica. En este caso no tenemos una función empírica, sino que un ideal de función[7] tal que minimiza nuestro criterio de error, maximizando su eficiencia. Si no poseemos una forma funcional empírica por la cual guiarnos, se propone la siguiente aplicación en el modelo de Beresteanu:

[7] De la misma manera en que se propone que en el modelo de regresión simple hay una línea que mejor ajusta los resultados de la población, y que se intenta estimar.

Proposición 1. Metodología de Optimización.

1. Utilizamos como función de activación la forma funcional de Beresteanu

$$f' = \phi_{r,j} + f_{HT} \tag{13}$$

   La función base f_{HT} (llamada función base) sobre la cual se aplican los cambios funcionales es la función Hyperbolic-Tangent (ver anexo).

2. En vez de aplicar la minimización del criterio de error de Beresteanu, integramos la optimización de $\phi_{r,j}$ dentro de la optimización de Ψ. Ya que el modelo original estima $\phi_{r,j}$ minimzando el error de la función analítica versus la función observada, en este caso estimamos $\phi_{r,j}$ de manera implícita en la optimización de Ψ, alterándola de forma tal que disminuya Ψ.

Si bien cualquier función se puede utilizar como base en vez de f_{HT} (supeditado a que sea contínua), la utilización de esta función en particular como base se explica al discutir el razonamiento del algoritmo de optimización. No obstante, el razonamiento de esta elección no es muy lejano al del propuesto por Beresteanu: en vez de construir la grilla a partir de la línea del eje, ésta se derivará de los valores de una función de activación conocida. Bajo el mismo argumento, el proceso de Beresteanu puede ser interpretado como utilizando una función $f(x) = 0\ \forall x$.

Una de las ventajas directas que se obtienen al mezclar ambas funciones es que la nueva función generada f' no está limitada en su recorrido, no así como ocurre tanto con f_{HT} como con el resto de las funciones tradicionales. Esta nueva función transformada mantiene su dominio sobre todos los reales, pero ahora tiene un recorrido también sobre todos los reales.

Desarrollo Matemático de la Optimización

Nuestra tarea es ahora integrar los procesos de optimización del algoritmo de Backpropagation, y aquél de Beresteanu.

La introducción de este factor no alterará el proceso de optimización general desarrollado anteriormente, ya que no afecta hasta el punto en donde, por la regla de la cadena, debemos comenzar a derivar la derivada de las funciones de activación. Dentro de la metodología del Algoritmo de Backpropagation, esto corresponde a intervenir la formulación del factor de sensibilidad de la ecuación (4).

Como se explica en Hagan, Demuth y Beale (1996), el factor s_i^m aplicado a actualizaciones de las matrices de actualización (o de pesos), corresponde a

$$\boldsymbol{W}^m(k+1) = \boldsymbol{W}^m(k) - \alpha \boldsymbol{s}^m (\boldsymbol{a}^{m-1})' \qquad (14)$$
$$\boldsymbol{b}^m(k+1) = \boldsymbol{b}^m(k) - \alpha \boldsymbol{s}^m$$

Pues bien, para lograr calcular cuál es el factor de sensibilidad $\boldsymbol{s}^m$, es necesario computar el Jacobiano:

$$\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m} \equiv \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{s^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{s^m}^m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_{s^m}^m} \end{bmatrix} \qquad (15)$$

En donde,

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial \left(\sum_{l=1}^{s^m} w_{i,l}^{m+1} a_l^m + b_l^{m+1} \right)}{\partial n_2^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m}$$
$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial f^m \left(n_j^m \right)}{\partial n_j^m} = w_{i,j}^{m+1} f^m \left(n_j^m \right)$$

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = w_{i,j}^{m+1} f^m\left(n_j^m\right) \tag{16}$$

El Jacobiano se puede expresar entonces como

$$\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m} = \boldsymbol{W}^{m+1}\boldsymbol{F}^m(\boldsymbol{n}^m) \tag{17}$$

Con

$$\boldsymbol{F}^m(\boldsymbol{n}^m) \equiv \begin{bmatrix} f^m(n_1^m) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & f^m\left(n_{S^m}^m\right) \end{bmatrix} \tag{18}$$

Al aplicarlo en nuestro factor de sensibilidad, obtenemos $\boldsymbol{s}^m$,

$$\boldsymbol{s}^m = \frac{\partial \hat{\Psi}}{\partial n_i^m} = \left(\frac{\partial \boldsymbol{n}^{m+1}}{\partial \boldsymbol{n}^m}\right)' \frac{\partial \hat{\Psi}}{\partial n_i^{m+1}} = [\boldsymbol{W}^{m+1}\boldsymbol{F}^m(\boldsymbol{n}^m)]' \frac{\partial \hat{\Psi}}{\partial n_i^{m+1}} = [\boldsymbol{W}^{m+1}\boldsymbol{F}^m(\boldsymbol{n}^m)]'\boldsymbol{s}^{m+1} \tag{19}$$

Esto simplifica el resultado del modelamiento debido a que este elemento, que es usado directamente en el proceso de actualización de los parámetros de NNs (ver ecuación 12), es susceptible de ser alterado en el factor $\boldsymbol{F}^m(\boldsymbol{n}^m)$, utilizando de forma directa la proposición (ecuación 13),

$$f' = \phi_{r,j} + f_{HT}$$

Es decir,

$$\boldsymbol{F}'^{m}(\boldsymbol{n}^{m}) \equiv \begin{bmatrix} f'^{m}(n_1^m) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & f'^{m}(n_{S^m}^m) \end{bmatrix}$$

El factor de actualización consistente con la Proposición I es finalmente:

$$\boldsymbol{s}'^{m} = [\boldsymbol{W}^{m+1}\boldsymbol{F}'^{m}(\boldsymbol{n}^{m})]'\,\boldsymbol{s}'^{m+1} \tag{20}$$

La utilización de este factor de actualización en una red en que las funciones de activación son modificaciones en base a la metodología de Beresteanu de una función hiperbolic tangent tradicional, se plantea que corregiría los problemas de especificación del modelo expuestos en la primera parte de la tesis, y al mismo tiempo libraría al investigador de tener que tomar decisiones a priori, en base a criterios poco definidos.

Algoritmo de Resolución

El modelamiento anterior omite una problemática que queda de manifiesto al momento de implementar esta solución: el cálculo de $\boldsymbol{s}'^{m}$ depende directamente de las derivadas de f', las que a su vez dependen de las restricciones de forma establecidas por el investigador, y representadas por la ecuación (7). El desarrollo de una forma funcional que sea consistentes tanto con el proceso optimizador original (representado por el método de Backpropagation) como con la metodología de Beresteanu supera tanto las capacidades de este autor como el objetivo de esta tesis, por lo que no será un punto a desarrollar.

Ello no impide que se presente una segunda proposición desde el punto de vista numérico para la resolución del problema:

Proposición 2. Algoritmo de Optimización Escalonado.

1. Se define un número ε de *epochs*[8].

2. Se optimiza el modelo neuronal de la forma tradicional:

   Al inicializar $f^{'} = \phi_{r,j} + f_{HT} = f_{HT}$. Es decir, todos los parámetros de $\phi_{r,j}$ son cero. Esto corresponde a la optimización normal.

3. Al llegar a ε *epochs*, se pausa la optimización de los ponderadores de la red, para optimizar $\phi_{r,j}$. Esto se realiza mediante el método de optimización de Beresteanu, utilizando directamente las restricciones de forma, pero cambiando la función objetivo de su modelo por la de nuestra red.

4. Esta nueva configuración funcional resultado del paso 3 se aplica al modelo para seguir con el proceso de optimización de la red por un nuevo número de ε epochs.

5. Se reestima el modelo de función de Beresteanu con el mismo procedimiento del paso 3. El resultado de esta optimización es $\hat{\phi}_{r,j}^{\varepsilon(p+1)}$. Donde p es la iteración En este caso el modelo de actualización de parámetros sigue el mismo esquema que el modelo de Backpropagation:

$$\tilde{\phi}_{r,j}^{\varepsilon(p+1)}(x) = \tilde{\phi}_{r,j}^{\varepsilon p}(x) + \alpha\hat{\phi}_{r,j}^{\varepsilon(p+1)}(x) \tag{21}$$

   En este caso el nuevo factor del modelo, en el punto j de la grilla, va a ser igual al punto en la iteración εp, $\tilde{\phi}^{\varepsilon p}$ más un porcentaje α del nuevo factor optimizado

[8] En la metodología tradicional, se contabiliza un *epoch* una vez que ya se han ejecutado una iteración con cada dato incluido en la base original. Así una optimización con la base completa es un *epoch*.

$\hat{\phi}$, con el objeto de disminuir la divergencia del estimador. Este factor es el mismo factor α de actualización del modelo de Backpropagation.

6. Repite puntos 3 y 4 hasta convergencia a un nivel de error aceptable.

Este proceso algorítmico adolece de una de las críticas hechas a la elección de la forma funcional: se debe escoger un ε, en vez de dejar al modelo que lo estime. La proposición en este caso es que el valor de ε escogido afecta dos factores: el nivel de convergencia/divergencia del modelo, y la rapidez de esta.

Podemos proporcionar el siguiente razonamiento para validar la elección de la forma base de la función de activación (Proposición 1). Debido a la manera en que el algoritmo de optimización resuelve el modelo, si no se modifica la forma funcional de Beresteanu $(f(x) = 0\ \forall x)$, el resultado del paso 2 del algoritmo presentado en la Proposición 2, sería una red sin capacidad predictiva. Esto sucede ya que no hay zonas de exitación neuronal para generar discernimiento en base a los datos. En otras palabras, para cualquier vector de inputs, y cualquier vector de ponderadores inicializados, el resultado de la función de activación sería invariante (igual a cero), y además de derivada cero, invalidando los fundamentos de Backpropagation.

Para asegurarnos que esto no sucede, en vez de partir en $f(x) = 0\ \forall x$, se puede hacer en $f(x) = f_{HT}(x)$, o en $f(x) = f_{LS}(x)$. Cualquiera sea la elección, si se utilizan funciones en principio compatibles con Backpropagation, no tendremos dicho problema. Adicionalmente, debe ser diferenciable y con dominio en todos los números reales, para asegurarse que, como en el proceso de Beresteanu, éste converge a valores óptimos de $\hat{\phi}$.

Como se aclaró anteriormente, no es el objetivo de esta tesis el de modelar completamente los aspectos teóricos de un problema de estas características, sino que evidenciar las ventajas y factibilidad de automatizar procesos financieros que utilizen NNs, e incentivar investigaciones o desarrollos posteriores en donde se logre desarrollar un modelo completo que resuelva esta problemática.

Sección III. Evidencia de efectos de la elección funcional

En consecuencia con los objetivos de este trabajo buscaremos evidencia de los efectos que tiene la elección de una u otra configuración funcional respecto de los resultado de la red.

Primero se demostrará que se pueden establecer métodos de verificación para una elección óptima, señalando criterios teóricos que definan la mejor capacidad de un modelo, los cuales puedan ser verificados empíricamente a costos razonables en términos de procesamiento. Segundo, y más importante, es que validará la hipótesis central de esta tesis de que el investigador está incurriendo en un riesgo importante al escoger la función de activación sin tomar en cuenta los efectos que esto pudiera tener en los resultados del modelo final.

Medida de Eficiencia

Para verificar la calidad de un modelo particular tenemos variadas medidas de eficiencia sobre las cuales es posible optimizar la red: el Error Medio Absoluto, Error Cuadrático Medio, Suma de los Errores Cuadráticos, etc. Para verificar las configuraciones funcionales más eficientes se utilizará, en concordancia con el proceso de optimización mostrado anteriormente, el Error Cuadrático Medio, el cual es altamente utilizado para verificar la consistencia de los resultados, de manera que se asegure la convergencia.

Modelamiento

Acotaremos el problema a determinar la eficiencia de las funciones de la familia logística. Ello implica abordar los problemas de estimación que se definirán más adelante mediante modelos que varían en su formulación sólo en la elección de la función de activación. Las opciones son una función Log-Sismoid (LS),

$$f(x) = \frac{1}{1 + e^{-n}}$$

o una función Hyperbolic Tangent (HT),

$$f(x) = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$$

Para probar que es posible desarrollar este análisis de forma consistente, se efectuará en 2 escenarios de modelamiento distintos:

1. Predicción de la formula de Probabilidad Normal, lo que corresponde a un caso más teórico, y claramente no lineal, y la

2. Predicción de precios diarios de una acción escogida al azar, en base a una base de corte transversal precios pasados y retorno del IPSA

Procedimiento

Se establece la siguiente metodología a utilizar:

1. Se estima el modelo utilizando ambas configuraciones funcionales, pero manteniendo el resto de los parámetros de configuración de la red y datos de entrenamiento constantes.

2. Se computa el valor del error cuadrático medio para ambas NNs: ECM^{LS}, y ECM^{HT}.

3. Se computa la razón de errores: $r = ECM^{HT} - ECM^{LS}$

4. Se repiten los pasos 1 al 4, 10000 veces, para construir una distribución de valores tanto de r, ECM^{HT} y ECM^{LS}.

El límite de epochs para cada modelo aumenta al doble de lo normalmente utilizado[9], para asegurarse de que el modelo converge, pese a las elecciones funcionales. Ello deja el límite de epochs en 200[10].

[9] En modelos feed-forward como los utilizados en esta investigación, paquetes de software de análisis de redes neuronales, como Matlab en su versión 2006b, tienen como límite 100 epochs. En versiones posteriores, gracias a mejoramientos en la eficiencia del algoritmo se ha aumentado el límite a 1000

Consistentemente con lo explicado durante el desarrollo de este trabajo, la hipótesis central es que no es irrelevante la elección de una función de activación. Esto se traduce en testear las hipótesis de que las distribuciones de las poblaciones de ECMs generadas no sean iguales. La evidencia de que tanto la distribución como la media o la varianza cambien validaría la hipótesis planteada.

Para testear nuestra hipótesis, se harán dos tests sobre la distribución de errores de cada modelo: de comparación de medias y de varianzas.

Previo a ambos tests se realizará un testeo de normalidad de la distribución, para identificar si es posible aplicar test de hipótesis tradicionales o no paramétricos. Para identificar la normalidad de los datos se utilizarán tanto la grafica de la densidad por quintil de la distribución versus la distribución normal (*qnorm* en Stata), y el test de Skewness/Kurtosis de Normalidad.

En el caso de que no se pueda rechazar la hipótesis de normalidad, utilizaremos los tests tradicionales de diferencia de medias, y diferencia de varianzas (basados en el estadístico t).

Si el test de diferencia de medias indica diferencias significativas en ambas distribuciones, habrá evidencia de que la capacidad de ajuste de la red es afectada por la selección de la función. Luego, en el caso normal, la diferencia entre los tests de una cola nos indica la

---

epochs, pero para mantener la congruencia del proceso de optimización tradicional de Backpropagation mostrado en las secciones anteriores, con el algoritmo utilizado en el software, nos remitiremos a la versión tradicional.

[10] Cada modelo se estima optimizando la respuesta de la red frente a los individuos de la muestra de entrenamiento. El límite de entrenamiento, en donde termina el algoritmo de optimización, se alcanza ya sea al conseguir el objetivo de optimización de ECM = 0, o alternativa al llegar a un número de iteraciones igual al límite de epochs (si la base tiene 500 datos, y el límite de epochs es de 200, ello implica una cantidad máxima de 100.000 iteraciones).

distribución con media mayor, y por lo tanto, cuál función es más efectiva en abordar el problema.

El test de diferencias en varianza indica además la eficiencia de la estimación. Una distribución de con un menor rango de variación es preferida, ceteris paribus, a una distribución con mayor variación. Este tipo de testeo nos permite identificar el grado de susceptibilidad de ser afectado por variaciones en los procesos aleatorios que se introduce en el modelo al elegir una forma funcional en particular.

Ahora bien, si encontramos que la distribución de las diferencias no es normal, se invalidan los tests anteriores, por lo que en reemplazo utilizaremos tests no paramétricos: el *Mann-Whitney rank sum test*, indica si la distribución de las muestras son idénticas, bajo la alternativa de que no lo son. Este test no permite determinar qué tipo de distribución tienen los datos, o si la diferencia en ellas es debido a la media, o tampoco la varianza. No obstante, es posible realizar el *Robust Test for Equality of Variance* de Brown y Forsythe (1974), que indica si las varianzas de las dos poblaciones son iguales. Este test cumple con el requisito de que no necesita demostrarse previamente que la población en estudio es normal.

En todos los casos, el rechazo de las hipótesis de igualdad (de distribuciones, medias o varianzas) indicará que los resultados de los modelos son efectivamente dependientes de la forma funcional escogida, lo que validará la hipótesis de la tesis.

Resultados

Modelo 1: Predicción Probabilidad Normal

El primer modelo en que testearemos nuestra hipótesis trata de predecir la probabilidad generada por la función normal estandarizada:

$$N(x_i) = \frac{1}{-\sqrt{2\pi\sigma^2}} exp\left(\frac{(x_i - \bar{x})^2}{2\sigma^2}\right) = \frac{1}{-\sqrt{2\pi}} exp\left(\frac{{x_i}^2}{2}\right)$$

Para aproximar esta forma funcional son se genera una base de 500 datos utilizando h, que es una variable aleatoria de valores entre 0 y 1.

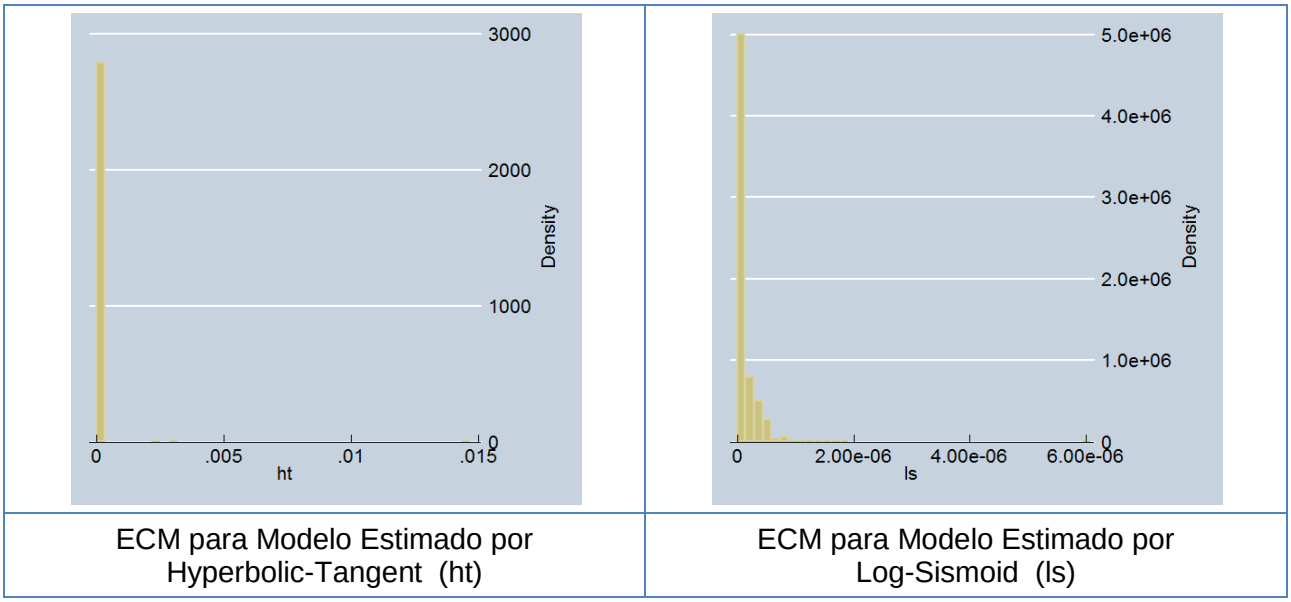
La red neuronal establecida para estimar estos valores corresponde a una red con 5 neuronas ocultas, y una neurona de salida.

Los quantile plots (ver en Anexo en página 37) se indican para las variables: **diff** corresponde a $r = ECM^{HT} - ECM^{LS}$, **ht** corresponde a la distribución de errores del modelo de NN estimado utilizando la función Hyperbolic-Tangent, y **ls** a la distribución de errores del modelo que utiliza Log-Sismoid.

Como queda en evidencia en los quantile plots, ninguna de estas distribuciones sigue un comportamiento normal. Se espera confirmar este resultado con el test de normalidad. Aún así, podemos ver que las distribuciones sobre la línea de normalidad para ht y ls son bastante diferentes no sólo en termino de escala (un rango mucho menor para ls que para ht), sino que también en forma. Ello ya nos da algún indicio débil de estar en presencia de distribuciones distintas.

Los valores reportados para todas las variables en los test de skewness y kurtosis son de un p-value de 0. Ello rechaza la nula de normalidad a cualquier significancia. Esto confirma que, pese a que el proceso de generación de datos fue un bootstrap experimental de 10.000 observaciones, la distribución de errores no es normal.

La observación de los siguientes histogramas demuestra la diferencia de las distribuciones, principalmente en varianza. La presencia de pocos pero sistemáticos valores extremos en el modelo **ht** elevan tanto la media como varianza de este modelo versus aquella de **ls**.



ECM para Modelo Estimado por Hyperbolic-Tangent (ht)	ECM para Modelo Estimado por Log-Sismoid (ls)

Gráfico 1 - Histogramas de ECM para NNs del Modelo 1 basadas en Hyperbolic-Tangent y Log-Sismoid.

	Test Normalidad		Rank-Sum Test		Paired Signed Rank Test		Robust Variance Test
Variable	P-Val Skew.	P-Val Kurt.	Rank Sum Index	P-Val	Rank Sum Index	P-Val	P-Val
Diferencia	0.00	0.00	-		-		
Hyperbolic-Tangent (ht)	0.00	0.00	0.775	0.0000	1.476	0.0000	0.2434
Log-Sismoid (ls)	0.00	0.00	1.225		0.524		

Tabla 1 - Resultados Tests Modelo Predicción Probabilidad Normal

Todos estos antecedentes invalidan la utilización de test tradicionales, por lo que usaremos el *Wilcoxon Rank-Sum Test* de Dos Muestras[11]. La hipótesis nula es que el rank-sum de ambas muestras es el mismo, lo que equivale a decir que ambas poblaciones tienen la misma función de distribución. Los resultados se reportan en la Tabla 1. Como el p-value de esta hipótesis es

[11] El rank sum index mostrado es una elaboración propia. Indica qué porcentaje del rank-sum esperado es obtenido.

cero, podemos rechazar la hipótesis nula a cualquier valor de significancia. Esto valida la hipótesis de la tesis, de que hay diferencias significativas entre los métodos utilizando diferentes funciones de activación.

Un testeo adicional que es posible de realizar es el *Two-Sample paired Signed Rank*. La hipótesis del modelo es acerca de la mediana de las diferencias (Northwestern University s.f.). Los resultados también se reportan en la Tabla 1. Como podemos ver también se rechaza que la mediana de las distribuciones sea la misma.

Un último test se realizó respecto a la varianza de las poblaciones, siguiendo la metodología de Brown y Forsythe (1974). Como vemos el p-value reportado es de 24.3%, lo cual impide rechazar la hipótesis a niveles estándar de significancia, pero es lo suficientemente baja como para tener indicios de que también difieren en varianza.

Modelo 2: Predicción Precio Acción COPEC

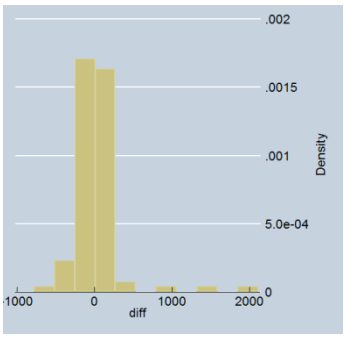
La configuración y objetivo de este modelo varían respecto del modelo anterior. En particular el objetivo es poder estimar el precio de una acción (se escogió Copec por ser la acción más líquida del período de estudio: los años 2000, 2001 y 2002). Para ello se armó un panel de datos que incluye una selección aleatoria (35%) de las acciones que en el período hayan tranzado más de 200 días. Al modelo se le ingresan los datos de los precios en $t = i$, para estimar el valor de los precios en $t = i + 1$.

La configuración de la red debe adecuarse, al tratarse de series de tiempo, por lo que en este modelo se diseña una red de tipo NARX con sólo un período de lag. En este caso hay feedback[12] de los datos en el vector de inputs (incluyendo los datos actuales y del período pasado), y desde el vector de outputs (con dos lags).

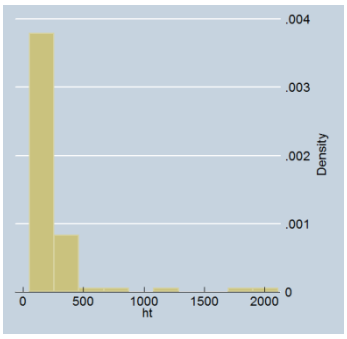
[12] (Hagan, Demuth y Beale 1996) discuten el diseño de este tipo de redes.

Si bien la configuración del modelo es distinta, aún es válida la especificación del algoritmo de optimización Backpropagation. En particular las redes NARX, a diferencia de las NARXSP, SP2NARX, o aquellas con capas recurrentes, no cambian en el proceso de optimización, por lo que el análisis de optimización aplica sin modificaciones.

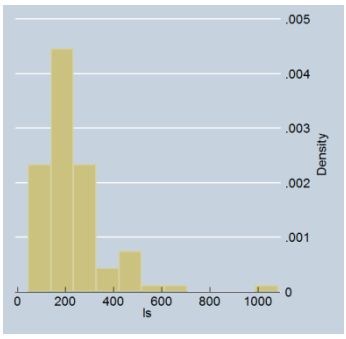
En este tipo de modelos, el tiempo que demora el algoritmo es muchísimo mayor al de los modelos tradicionales. Por esta razón el bootstrap se elaboró con una muestra reducida de modelos respecto al caso anterior: son 1054 modelos, lo cual aún así es una gran cantidad de datos. En este caso, aunque la cantidad de veces que se repite el experimento es menor, los datos están mucho mejor comportados, lo que queda de evidencia en los histogramas:



Diferencia en ECMs para ambos modelos (ht – ls)



ECM para Modelo Estimado por Hyperbolic-Tangent (ht)



ECM para Modelo Estimado por Log-Sismoid (ls)

Gráfico 2 - Histogramas de ECM para NNs del Modelo 2 basadas en Hyperbolic-Tangent y Log-Sismoid, y su diferencia.

Aún así, esta mejora en el comportamiento de los datos no es indicativa del comportamiento de una función normal, por lo que en este caso tampoco es posible hacer uso de tests tradicionales. Esto queda en evidencia en los quantile plots reportados en el anexo 2.a, y en el test de normalidad reportado en la Tabla 2.

	Test Normalidad		Rank-Sum Test		Paired Signed Rank Test		Robust Variance Test
Variable	P-Val Skew.	P-Val Kurt.	Rank Sum Index	P-Val	Rank Sum Index	P-Val	P-Val
Diferencia	0.00	0.00	-		-		
Hyperbolic-Tangent (ht)	0.00	0.00	1.065	0.0000	0.858	0.0001	0.1686
Log-Sismoid (ls)	0.00	0.00	0.935		1.142		

Tabla 2 - Resultados Tests Modelo Predicción Precio COPEC

El *Wilcoxon Rank-Sum Test* de Dos Muestras no tiene diferencias respecto del caso anterior. En esta ocasión nuevamente existe evidencia suficiente para rechazar la hipótesis nula de que el rank-sum de ambas muestras es el mismo. Este hecho da luz de un importante resultado de nuestro modelamiento: demostramos que la selección de la función de activación efectivamente produce diferencias en los resultados, y esta diferencia es robusta respecto al tipo de problema evaluado. Como podemos ver en el *Two-Sample paired Signed Rank Test,* los resultados del modelo también permiten rechazar que la mediana de las distribuciones sea la misma, y el testeo sobre la varianza, que no asume una distribución de origen (Brown y Forsythe 1974), tiene un p-value más favorable que el del caso anterior, de un 16.9% versus el 24.4% del modelo anterior.

Comparativa

Ya que hemos revisado los resultados de los modelos por separado, para obtener una visión global de lo encontrado podemos recapitular que:

a. En ambos escenarios, aquél de predicción de la probabilidad normal y el de predicción de precios accionarios de COPEC, tanto para el modelo estimado por Hyperbolic-Tangent como para el estimado por Log-Sismoid, se demostró que la distribución del ECM no es normal.

b. Al comparar dentro de cada escenario si la distribución del modelo resuelto por Hyperbolic-Tangent versus aquél resuelto por Log-Sismoid eran iguales se encontró con alta significancia que no es así.

c. Nuevamente para ambos escenarios, se encontró evidencia para rechazar que la mediana de los ECMs no es la misma en ambos tipos de modelamiento, y si bien no se pudo determinar lo mismo en el caso de las varianzas, los niveles de significancia a los que son diferentes, no son en extremo altos.

Un comparativo de los resultados que validan esta comparativa se entrega en el anexo de la página 39.

Con esta evidencia podemos demostrar que:

a. Utilizando una metodología bastante básica como la mencionada en esta tesis, el verificar si la elección de una función de activación es óptima no es costoso, versus el riesgo de la alternativa de confiar en el juicio del investigador

b. Efectivamente hay un efecto sobre la medida de eficiencia (en este caso el ECM) al elegir una u otra función de activación.

c. Los resultados son robustos al cambiar el escenario de estimación.

Mejoras en los resultados

A continuación se presenta la media del ECM de los modelos estimados en ambos escenarios.

	diff	ht	ls
Prediccion Función Normal Acumulada	2.09E-06	2.19E-06	1.05E-07
Predicción Precios Accionarios COPEC	81.70	332.64	250.93

Tabla 3 - Media de las Variables estudiadas para Ambos Modelamientos

En el caso del primer problema de estimación la diferencia es muy marcada, ya que el ECM del modelo estimado por Hyperbolic-Tangent es casi 21 veces el error del modelo estimado por Log-Sismoid. En términos prácticos, si estuvieramos hablando de la definición de probabilidad de impagos, por ejemplo, un investigador que equivocadamente hubiese escogido estimar por log-sismoid, podría disminuir en un 95.2% el error cuadrático del modelo al aplicar nuestra metodología. Esto refina la capacidad del modelo, que con cualquier función ya es muy buena, llevándola de una precisión promedio en el sexto decimal, a una precisión promedio de siete decimales.

Para el segundo problema de estimación, las mejoras por la estimación realizada por Log-Sismoid implican un descenso del error de estimación de un 24.6%, lo que equivale a $81.7 pesos de error promedio menos que en el caso estimado por Hyperbolic-Tangent.

Si bien las mejoras no son consistentes en diferentes escenarios, ya que influyen la escala de los datos, la dificultad del problema de estimación, y la capacidad de la red propuesta para enfrentarlo (aquellos otros parámetros que se explicó en un comienzo eran elegidos a mano por el investigador), vemos que en ambos casos sí hay una mejora, y que en el peor de los casos la metodología es capaz de disminuir en un 20% el error del modelo. Eso quiere decir que los valores predichos están un 20% más cerca de los valores reales al aplicar la metodología.

Sección IV. Conclusiones

Como quedó demostrado, el tomar una decisión respecto de una función de activación o de otra no es gratuito. Tiene efectos en aumentar los niveles de error en hasta 21 veces, y además altera la media y varianza de los errores. Es relevante que no pudimos encontrar evidencia que rechace la suposición de que estos efectos son generales, ya que como se mostró testeamos el modelo en dos contextos muy dispares, sin encontrar mayores diferencias entre ambos: en ambos hay evidencia estadísticamente significantes de que optimizar la NN mediante funciones diferentes afecta los resultados. Por ello hay un riesgo inherente en el dejar este factor sin un análisis para guiar la selección.

Esto deja de manifiesto la posibilidad de mejorar el método de optimización de las NNs, eliminando el sesgo introducido por el factor humano que representa la decisión poco informada del investigador. Como se mostró, el realizar estos testeos no es costoso en términos de capacidad computacional[13].

Se delinearon además ciertos procedimientos que dejan las bases para el desarrollo de una teoría que permita optimizar la selección funcional, del punto de vista de la creación de una función por medio de una grilla de puntos. Si bien el desarrollo teórico es bastante modesto, y carece de una mayor profundidad que deje a un modelo de estas características en condiciones de ser funcional, tanto académica como comercialmente, es de esperar que posteriores investigaciones desarrollen dicho modelo. Asimismo, es posible que nuevos avances en el campo en la medida de que se investigue este tema entreguen nuevas evidencias de los efectos de ciertas elecciones, de la misma manera en como en la literatura han surgido evidencias de que ciertas configuraciones de la red tienen mayor o menor efectividad en ciertos tipos de problemas.

[13] Todos los modelos y testeos fueron elaborados en un Notebook personal, de mucho menores prestaciones que los equipos de investigación profesionales

Anexos

Anexo 1. Esquematización Funciones Comunes

Función	Relación Matemática
Hard Limit Transfer Function	$f(x) = \begin{cases} a = 0 & n < 0 \\ a = 1 & n \geq 0 \end{cases}$
Symetrical Hard Limit	$f(x) = \begin{cases} a = -1 & n < 0 \\ a = 1 & n \geq 0 \end{cases}$
Linear Transfer Function	$f(x) = x;$
Saturating Linear	$f(x) = \begin{cases} a = 0 & n < 0 \\ a = n & 0 \leq n \leq 1 \\ a = 1 & n > 1 \end{cases};$
Symmetric Saturating Linear	$f(x) = \begin{cases} a = -1 & n < -1 \\ a = n & -1 \leq n \leq 1 \\ a = 1 & n > 1 \end{cases};$
Log-Sismoid Transfer Function	$f(x) = \frac{1}{1+e^{-n}};$
Hyperbolic Tangent Sigmoid	$f(x) = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}};$
Positive Linear	$f(x) = \begin{cases} a = 0 & n < 0 \\ a = n & n \geq 0 \end{cases}$
Competitive	$f(x) = \begin{cases} a = 1 & x = max\{x_i\}\ \forall i \\ a = 0 & x \neq max\{x_i\}\ \forall i \end{cases}$

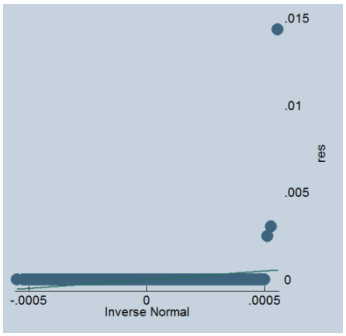
Tabla 4 - Esquematización de Funciones Comunes

Anexo 2. Tests de Normalidad

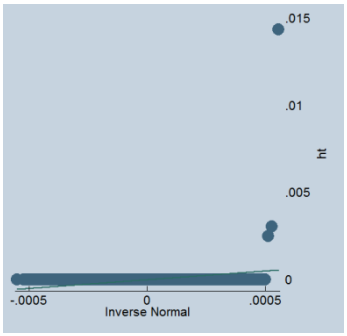
Los quantile plots son una forma de verificar si la distribución de una muestra es normal. Para ello se grafican los quantiles de la muestra (puntos azules) versus aquellos de una distribución normal (representada por una línea azul contínua). Mientras menos diferencias hayan entre los puntos y la línea, la distribución de la muestra se acerca a una distribución normal.

Modelo 1: Predicción Distribución Normal

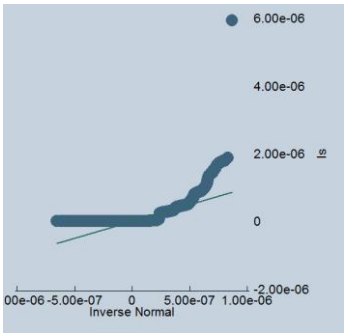
Quantile Plots



diff



ht



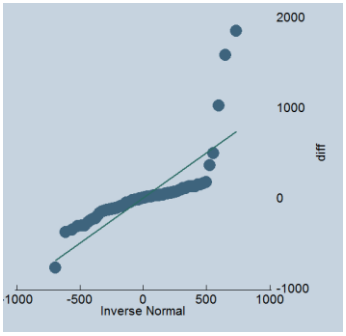
ls

Gráfico 3 - Quantile Plots para los ECM, diferencia, Hiperbolic-Tangent y Log-Sismoid del Modelo I

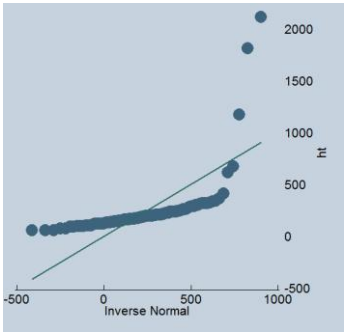
La presencia de fuertes diferencias al extremo derecho de cada gráfico, y tambibén la diferencia de pendientes en la zona en que ambos gráficos son cercanos implican que las distribuciones no son normales, para diff y ht. En el caso de ls, también se verifica que la distribución no es normal.

Modelo 2: Predicción de Precios Accionarios de COPEC

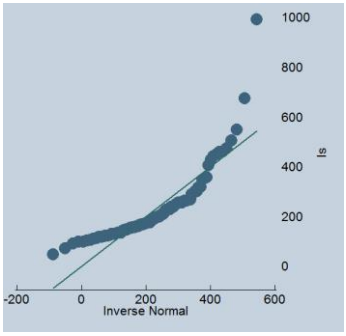
Quantile Plots



Diff



ht



ls

Gráfico 4 - Quantile Plots para los ECM, diferencia, Hiperbolic-Tangent y Log-Sismoid del Modelo II

Es claro que los puntos no siguen la línea recta, y por lo tanto la interpretación es que las poblaciones no son normales.

Anexo 3. Comparativa Resultados Tests para ambos Modelos

		Test Normalidad		Rank-Sum Test	
Modelo	Variable	Pr(Skewness)	Pr(Kurtosis)	Rank Sum Index	p-value
Modelo 1	Diferencia (diff)	0.00	0.00	-	0.0000
	Hyperbolic-Tangent (ht)	0.00	0.00	0.775	
	Log-Sismoid (ls)	0.00	0.00	1.225	
Modelo 2	Diferencia	0.00	0.00	-	0.0000
	Hyperbolic-Tangent (ht)	0.00	0.00	1.065	
	Log-Sismoid (ls)	0.00	0.00	0.935	

		Paired Signed Rank Test		Robust Variance Test
Modelo	Variable	Rank Sum Index	p-value	p-value
Modelo 1	Diferencia (diff)	-	0.0000	0.2434
	Hyperbolic-Tangent (ht)	1.476		
	Log-Sismoid (ls)	0.524		
Modelo 2	Diferencia	-	0.0001	0.1686
	Hyperbolic-Tangent (ht)	0.858		
	Log-Sismoid (ls)	1.142		

En los dos escenarios investigados: el Modelo 1, basado en datos de la función Normal Acumulada, y el Modelo 2, basado en datos accionarios para la predicción de precios de Copec, para las 3 variables estudiadas, el ECM de la Hyperbolic-Tangent (ht), aquél de la Log-Sismoid (ls), y su diferencia (diff), se testeó lo siguiente:

a. Normalidad. En todos los casos, los p-values son iguales a cero, lo que rechaza la hipótesis nula de que la distribución de dichas variables es normal.

b. Rank-Sum test. Realizado sobre ht versus ls en cada modelo: el p-value rechaza la hipótesis de que las distribuciones son iguales.

c. Paired Signed Rank Test. Realizado sobre ht versus ls en cada modelo: el p-value rechaza la hipótesis de que las distribuciones tienen la misma mediana.

d. Robust Variance Test. Realizado sobre ht versus ls en cada modelo: el p-value no puede rechazar a niveles de significancia convencionales (hasta 10%) la hipótesis de que las varianzas son diferentes.

Como conclusión, vemos que hay suficiente evidencia para asegurar que los resultados del modelo evaluados sobre su ECM varían considerablemente. Ello valida la hipótesis de que la decisión de utilizar una función en particular afecta los resultados del modelo de NNs.

Bibliografía

Aït-Sahalia, Y, y J. Duarte. «Nonparamentric option pricing under shape restrictions.» *Journal of Econometrics*, nº 116 (2003): 9-47.

Beresteanu, Arie. «Nonparametric estimation of regression functions under restrictions on partial derivatives.» *Versión Preliminar*, 2007.

Block, Ryan. *Engadget.* Septiembre de 2004. http://www.engadget.com/2004/09/27/aston-martin-db9-engines-get-neural-network.

Brent, R.P. *Algorithms for minimization without derivatives.* Englewood Cliffs: Prentice Hall, 1973.

Brown, M., y A. Forsythe. «Robust test for the equality of variances.» *Journal of the American Statistical Association* 69 (1974): 364-367.

Chen, X., J. Racine, y N. Swanson. «Semiparametric ARX Neural Network Models with an Application to Forecasting Inflation.» *Preliminary Version*, 2001.

Colin, A. M. «Neural Networks and Genetic Algorithms for Exchange Rate Forecasting.» *Proceedings of International Joint Conference on Neural Networks*, Noviembre 1992.

Demuth, H., M. Beale, y M. Hagan. *Neural Network Toolbox 6.* Edición Online. Vol. Octubre 2008. Mathworks, 2008.

Dryden Flight Research Center, NASA. *IFCS Fact Sheet.* Junio de 2006. http://www.nasa.gov/centers/dryden/news/FactSheets/FS-076-DFRC.html.

González, I., y J. M. Jiménez. *Predicción de la variación del tipo de cambio con redes neuronales: rolling versus recursividad.* Santiago: Tesis para optar al título de Ingeniero Comercial, Universidad de Chile, 2003.

Gutiérrez, M. *Administración de Carteras con redes neuronales mediante metodología Rolling.* Santiago: Tesis para optar al título de Ingeniero Comercial, Universidad de Chile., 2004.

Hagan, M., H. Demuth, y M. Beale. *Neural Network Design.* PWS Publishing, Thomson Learning, 1996.

Hull, J. *Options, Futures and Other Derivatives.* 6th Edition. New Jersey: Pearson Education, 2006.

Leigh, W., R. Purvis, y Ragusa J. M. «Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural networks, and genetic algorithm: a case study in romantic decision support.» *Decision Support Systems* 32, nº 4 (2002): 361 - 377.

McCulloch, W., and W. Pitts. "A logical calculus of the ideas immanent in nervous activity." *Bulletin of Mathematical Biophysics* 5 (1943): 115-133.

McNelis, P. *Neural Networks in Finance: Gaining Predictive Edge in the Market.* Burlintong, MA: Elsevier Academic Press, 2005.

Mutihac, R., y M. Van Hulle. «A comparative survey on adaptive neural networks algorithms for independent component analysis.» *Romanian Reports in Physics* 55, nº 1 (2003): 43-67.

Northwestern University. «Propeth STAT Guide.» *Nonparametric Tests.* http;//www.basic.northwestern.edu/statguidefiles/nonpar.html (último acceso: Octubre de 2008).

Peltarion. *Application of Adaptive Systems.* http://www.peltarion.com/doc/index.php?title=Applications_of_adaptive_systems.

Rosenblatt, F. «The perceptron: A probabilistic model for information storage and organization in the brain.» *Psychological Review* 65 (1958): 386-408.

Tsai, C., y J. Wu. «Using neural network ensembles for bankruptcy prediction and credit scoring.» *Expert Systems with Applications: An International Journal* 34, nº 4 (2008).