Predicción del impuesto a la renta usando Multi-Layer perceptrons con el algoritmo de entrenamiento de Levenberg-Marquardt

Paolo Maldonado Hurtado
Ingeniería de TI y Sistemas
Universidad ESAN
Lima, Perú
17200822@ue.edu.pe

***Abstract*—En este proyecto se predijo el impuesto a la renta del año 2020 recaudado por la Superintendencia Nacional de Aduanas y de Administración Tributaria - SUNAT usando datos históricos de una serie temporal mensual desde el 2005 hasta el 2020. Se experimentó con varios multi-layer perceptron usando búsqueda exhaustiva para encontrar la mejor arquitectura de red y se entrenó con el algoritmo de Levenberg-Marquardt. El modelo se optimizó usando mas repeticiones obteniendo resultados favorables para el forecasting.**

***Index Terms*—multi-layer perceptron, forecasting, Levenberg-Marquardt, impuesto a la renta**

I. Introducción

El impuesto a la renta es un tributo que se determina anualmente, tiene vigencia del 01 de enero al 31 de diciembre. Se aplica a una persona natural cuando sus ingresos que provienen del arrendamiento u otro tipo de cesión de bienes muebles o inmuebles. También del trabajo realizado de forma dependiente o independiente, siempre que no se realice actividad empresarial.
La Superintendencia Nacional de Aduanas y de Administración Tributaria - SUNAT tiene como finalidad primordial administrar los tributos del gobierno nacional y los conceptos tributarios. En este contexto, los impuestos recaudados por la SUNAT se hacen mes a mes y resultaría útil usar alguna herramienta de forecasting que permite predecir estos valores a futuro de modo que este organismo podría organizarse mejor y prever ciertos escenarios gracias a información futura, las redes neuronales también son útiles en el ámbito del forecasting por lo que en este trabajo se plantea usar una red neuronal para predecir el impuesto a la renta.

II. Conjunto de datos

La base de datos usada para predecir el impuesto a la renta contiene información de los impuestos recaudados de forma mensual por la SUNAT a manera de serie temporal y va desde enero del 2005 hasta diciembre de 2020 comprendiendo un total de 192 meses.

En cuanto al preprocesamiento de datos, se normalizó cada valor mensual en el rango de [-1, 1] usando la ecuación (1).

$$y_i = \frac{(y_{max} - y_{min}) * (x_i - x_{min})}{(x_{max} - x_{min})} + y_{min} \quad (1)$$

Donde y_i representa el dato normalizado entre el rango definido en (1) y x_i el valor i de la serie de datos

III. Modelo

Se exploraron varios modelos multi-layer perceptron siguiendo una misma arquitectura en busca de encontrar la mejor combinación de variables de entrada y neuronas en la capa oculta, para ello se empleo el algoritmo de búsqueda por fuerza bruta o también conocido como búsqueda exhaustiva.

A. Variables de entrada

Se contó originalmente con 7 variables de entrada las cuales se detallan en la tabla (1). El número de neuronas en la capa de entrada fue variando desde tener subconjuntos de 2 entradas hasta 7 entradas.

TABLA I
Variables de entrada

Promedios móviles	**Retrasos de valor**					
MM2	M-1	M-2	M-3	M-4	M-6	M-12

Detalle de variables originales para entrenar el modelo

El modelo original con 7 entradas puede resumirse de acuerdo con la ecuación (2).

$$Y = f(MM2, S_{t-1}, S_{t-2}, S_{t-3}, S_{t-4}, S_{t-6}, S_{t-12}) \quad (2)$$

$MM2 = \frac{S_{n-1} - S_n}{2}$

B. Función de activación

Se utilizó la función de activación tangente hiperbólica (Tansig) para la capa oculta y la función de transferencia lineal (Purelin) para la capa de salida.

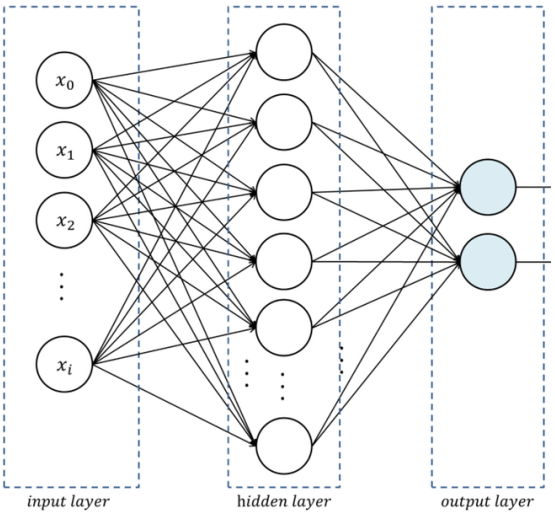

Fig. 1. Arquitectura referencial del modelo multi-layer perceptron con 1 capa de entrada, 1 capa oculta y 1 capa de salida

C. Evaluación del desempeño

Para la evaluación se utilizó el error porcentual absoluto medio (MAPE) de modo que el mejor modelo resultante de los experimentos es el que tenga menor MAPE. Para hallar esta medición se utilizo lo mostrado en la ecuación (3).

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{3}$$

n =número de observaciones
A_t =valor actual
F_t =valor pronosticado

IV. Experimentos

A. Búsqueda exhaustiva

Se usó el algoritmo de búsqueda exhaustiva para determinar la mejor combinación de entradas y cantidad de neuronas en la capa oculta. Para esto, se experimentó con subconjuntos de 2 hasta 7 entradas, cada uno con su propia cantidad de combinaciones posibles. Por ejemplo, en un subconjunto de 2 entradas hay $\binom{7}{2}$ combinaciones o 21, entonces mediante fuerza bruta se van eligiendo cada una de estas 21 combinaciones para servir de entrada al modelo.
Asimismo, cada combinación en la capa de entrada fue probada con diferentes cantidades de neuronas en la capa oculta, cantidad que variaba de entre 1 a 7 neuronas. Se uso el algoritmo de Levenberg-Marquardt para entrenar cada arquitectura resultante de las combinaciones con un total de 25 repeticiones por cada proceso total terminado y un multi-step de 12.

El mejor modelo fue determinado por el menor MAPE, la tabla (2) muestra un resumen de la mejor arquitectura por cada subconjunto de entrada. De allí se puede observar que el menor error lo tiene la arquitectura [4 6 1] con un 17.0231%, es decir, 4 neuronas en la capa de entrada, 6 en la capa oculta y 1 en la capa de salida.

TABLA II
Comparación de mejores arquitecturas

Arquitectura	Entradas	MAPE
[2 6 1]	$MM2$, S_{t-6}	19.9383%
[3 6 1]	S_{t-1}, S_{t-6}, S_{t-12}	17.3863%
[4 6 1]	S_{t-1}, S_{t-4}, S_{t-6}, S_{t-12}	17.0231%
[5 4 1]	$MM2$, S_{t-1}, S_{t-3}, S_{t-6}, S_{t-12}	18.0605%
[6 2 1]	S_{t-1}, S_{t-2}, S_{t-3}, S_{t-4}, S_{t-6}, S_{t-12}	18.0041%
[7 3 1]	$MM2$, S_{t-1}, S_{t-2}, S_{t-3}, S_{t-4}, S_{t-6}, S_{t-12}	18.0160%

B. Modelo final

Para optimizar el modelo obtenido de la búsqueda exhaustiva, este se volvió a entrenar bajo el algoritmo de Levenberg-Marquardt y esta vez con 500 repeticiones obtenido los resultados que se muestran en la tabla (3). Los pesos y el bias se obtuvieron mediante la función de descenso de gradiente con momentum (learngdm) y se muestran en las tablas (4) y (5) respectivamente.

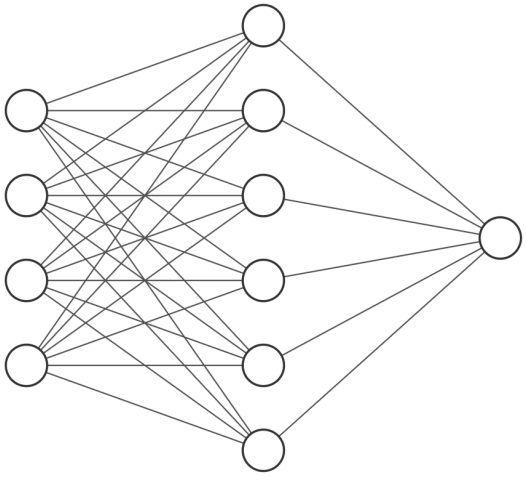

Fig. 2. Modelo final con arquitectura [4 6 1] del modelo multi-layer perceptron

TABLA III
Modelo final optimizado

Algoritmo	Arquitectura	Entradas	MAPE
trainlm	[4 6 1]	S_{t-1}, S_{t-4}, S_{t-6}, S_{t-12}	7.5831%

V. Resultados y conclusiones

Con el modelo obtenido se realizo la predicción del impuesto a la renta para el año 2020 y este resultado se contrastó con los datos verdaderos pertenecientes a la SUNAT de ese mismo año. El resultado se puede observar en la figura (3).

TABLA IV
PESOS Y BIAS W_1 DEL MODELO

Bias θ	x1	x2	x3	x4
-4.3610	12.70638	-1.76880	1.90555	-0.29867
-5.1485	15.78111	0.14813	1.17838	0.31945
0.8029	-1.53815	-0.22693	-0.02943	2.54100
-3.0700	7.13881	-1.59776	0.09707	-0.37135
-1.2413	-2.82290	0.76295	1.03141	-0.38606
-1.7650	-0.97004	1.53796	1.71019	-3.52745

TABLA V
PESOS Y BIAS W_2 DEL MODELO

Bias	x1	x2	x3	x4	x5	x6
-0.2151	-4.0581	2.2006	0.6976	2.2525	-0.7956	0.5786

Se puede concluir que el modelo utilizado ha sido capaz de predecir algunos valores con mas precisión que otros, por ejemplo en la figura (3) se observa que hay una diferencia notoria entre lo predicho y lo real en los meses de abril y junio, mientras que en los demás meses el modelo ha predicho casi correctamente; algunas mejoras clave identificadas para poderse trabajar en un futuro consisten en que se podría incrementar el numero de repeticiones al momento de entrenar y experimentar con otros algoritmos de entrenamiento y otras funciones de activación.

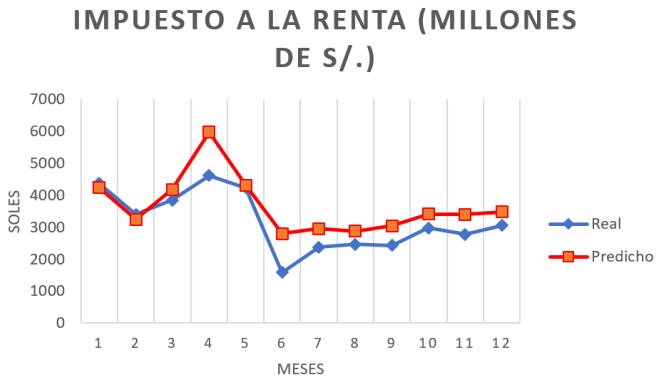

Fig. 3. Contraste entre los valores reales y predichos

VI. REFERENCIAS

SUNAT. https://www.sunat.gob.pe/institucional/quienessomos/ Gobierno del Perú. https://www.gob.pe/664-impuesto-a-la-renta