MODELO DE REGRESIÓN PARA LA PRODUCCIÓN DEL BRÓCOLI USANDO REDES NEURONALES ARTIFICIALES

REGRESSION MODEL BY AN ARTIFICIAL NEURAL NETWORK WITH MACROS IN EXCEL FOR THE BROCCOLI CROP YIELD

Oscar Francisco González Ramírez
Tecnológico Nacional de México / ITS Lerdo, México
mm2210886@itslerdo.edu.mx

Raymundo Juárez Del Toro
Universidad Autónoma de Coahuila / FCA Torreón, México.
r.juarez@uadec.edu.mx

Francisco Ruvalcaba Granados
Tecnológico Nacional de México / ITS Lerdo, México
francisco.rg@itslerdo.edu.mx

José Cruz Olvera Ávila
Tecnológico Nacional de México / ITS Lerdo, México
mm2210984@itslerdo.edu.mx

Recepción: 27/febrero/2023 **Aceptación:** 15/septiembre/2023

Resumen

El uso de Redes Neuronales Artificiales (RNA) en problemas de predicción de variables es cada vez más común. En la agricultura se generan continuamente una gran cantidad de datos de clima, biometría de la planta, días de cosecha y producción. Los lenguajes de programación o software especializado ya incorporan cada vez más las herramientas y algoritmos del Aprendizaje Automático (ML). Desde un punto de vista educativo o de implementación inicial básica y simple, el acceso a estos programas puede ser difícil. Por otro lado, las hojas de cálculo como Excel® incorporan utilidades de programación, como las macros en Visual Basic, que permiten implementar las RNA de una manera accesible para todos. Entre estas herramientas están las MACROs programadas desde Visual Basic, que se activan habilitando el complemento de programador en Excel. El objetivo de este trabajo es la creación de una MACRO en Excel que permita estimar un modelo de regresión para el ajuste de los valores del rendimiento de la producción del brócoli en una

granja agrícola local en Aguascalientes, México por lo cual en el presente documento se podrá observar el funcionamiento de dicho modelo y verificar la forma de trabajo del algoritmo mediante el entrenamiento y prueba de la RNA utilizando una base de datos del año 2019. Entonces, los valores de los pesos obtenidos constituyen el punto de partida para la validación de la RNA con los datos de cosecha del año 2023.

Palabras Clave: Agricultura de precisión, Red Neuronal Artificial, Macros de Excel.

Abstract

The use of Artificial Neural Networks (ANN) in forecasting problems is becoming more common. In agriculture, a large amount of data on weather, biometrics of the plant, days of harvest and production are continuously generated. Programming languages or specialized software incorporate more and more tools that allow the implementation of Machine Learning (ML) algorithms. From an educational or basic business implementation point of view, access to these programs can be difficult. On the other hand, spreadsheets such as Excel® incorporate utilities that allow ANN to be implemented and are easily accessible to everyone. Among these tools are the MACROS programmed from Visual Basic, enabling the programmer add-in in Excel. The objective of this work is the creation of a MACRO in Excel that allows estimating a regression model for the adjustment of the values of the yield of broccoli production in a local agricultural farm in Aguascalientes, Mexico, for which in this document You will be able to observe the operation of said model and verify the way the algorithm works by training and testing the ANN using a database from 2019. Then, the values of the weights obtained constitute the starting point for the validation of the ANN with the harvest data of the year 2023.

Keywords**:** *Artificial Neural Network; Agriculture automation; Excel macros.*

1. Introducción

Las Redes Neuronales Artificiales (RNA) son una herramienta del Aprendizaje Automático (ML) y su uso se ha convertido en algo común en distintas disciplinas; es por esta razón que se requiere de ejemplos numéricos y tutoriales básicos y

simples sobre la implementación de este tipo de herramientas en la temática diaria de una hoja de cálculo, para que el análisis de datos esté disponible en software de uso común. Si bien existe una gran cantidad de software especializado, como MATLAB, STATISTICA, SPSS; así como en lenguajes de programación como Python, R, C++, entre otros; que ya incorporan módulos de RNA en sus herramientas de análisis, dicho software no es de uso común para los estudiantes e incluso para las empresas, además de que funcionan como una caja negra y no facilita la comprensión de las RNA. En este trabajo se pretende dar una explicación clara y sencilla de la implementación de una RNA a través de la programación de una MACRO en una hoja de cálculo de Excel® esto mediante el uso de fórmulas y algoritmos predefinidos que ayudaran a llegar al objetivo planteado. El uso de hojas de cálculo permite al estudiante y a cualquier persona entender y apropiarse del funcionamiento de una RNA, facilitando el acceso a una metodología cada vez más extendida y que puede utilizar para resolver algún problema específico en su entorno común sin embargo para poder aplicarlos implica que la persona que desee implementarlos tenga conocimientos básicos acerca del uso de este tipo de algoritmos. Algunos trabajos que han contribuido al objetivo de implementar una RNA en herramientas tan comunes como la hoja de cálculo and sido por ejemplo García (2002), que utiliza una hoja de cálculo para la simulación de una RNA; también Segee y Amos (1997) ya tenían implementaciones de este tipo en Windows 95; Demir et al. (2018) aplicaron una RNA con propagación hacia atrás para la educación de estudiantes de ingeniería; y finalmente Semerikov et al. (2020) que también utilizaron las hojas de cálculo para la implementación de algoritmos de aprendizaje automático. Ver ambos trabajos en [3], [4], [5] y [6]. En este trabajo esa aplicación es el entrenamiento y prueba de la RNA para encontrar un modelo de ajuste que describa el comportamiento del rendimiento de la producción en la cosecha del Brócoli, en una granja agrícola local en Aguascalientes, México. Este modelo de ajuste se hará en base a ciertas variables meteorológicas tales como temperaturas, precipitación y radiación, registradas durante el año 2019.Se tomaron en cuenta las variables anteriormente mencionadas ya que se determinó que son las que más afectan al desarrollo de las plantas de brócoli y que la información

obtenida en base a ellas ayudarían al entrenamiento de la red. Además, la importancia de esta implementación es la de utilizar el modelo obtenido para validar los nuevos datos de cosecha, a partir de las variables meteorológicas para el presente año 2023.

Las redes neuronales son elementos de cálculo inspirados en las redes de neuronas biológicas. Como estas últimas, están constituidas por elementos simples denominados nodos o neuronas organizados en capas e interconectados. A la forma particular de organizarse y conectarse la neurona se le denomina arquitectura. A cada conexión se le asigna un peso numérico w_{ij}, que va a constituir el principal recurso de memoria a largo plazo de la red. El aprendizaje se realiza, usualmente, con la actualización de los pesos mediante una determinada regla de aprendizaje. Las neuronas se suelen organizar en columnas paralelas denominadas capas. Existen tres tipos de capas, de entrada, de salida y ocultas. Las neuronas de la capa de entrada reciben señales del entorno y no realizan más operación que la de distribuir estas señales a las neuronas de la capa siguiente. Las neuronas de la capa de salida procesan la información y envían la señal fuera de la red. Las unidades ocultas son aquellas cuyas entradas y salidas se encuentran dentro de la red. Las neuronas de capas distintas pueden estar conectadas unidireccional o bidireccionalmente. Las redes que sólo contienen relaciones unidireccionales se denominan estáticas o redes acíclicas. Las redes con relaciones bidireccionales se denominan redes dinámicas o cíclicas. En las redes estáticas la información fluye en un solo sentido (ni lateralmente ni hacia atrás). Entre las redes dinámicas se pueden distinguir las que contienen relaciones bidireccionales entre los nodos de la misma capa, denominadas redes competitivas, y redes con relaciones bidireccionales entre neuronas de distintas capas, denominadas redes recurrentes. A su vez, las neuronas de distintas capas pueden estar total o parcialmente conectadas. A las redes con conexiones unidireccionales y totalmente conectadas se les denomina perceptrones multicapa o redes feedforward y son, junto a las redes competitivas, las más utilizadas en análisis de datos. La figura 1 muestra el esquema de una red neuronal con *n* neuronas en la capa de entrada, *m* neuronas en la capa oculta y una neurona en la capa de salida.

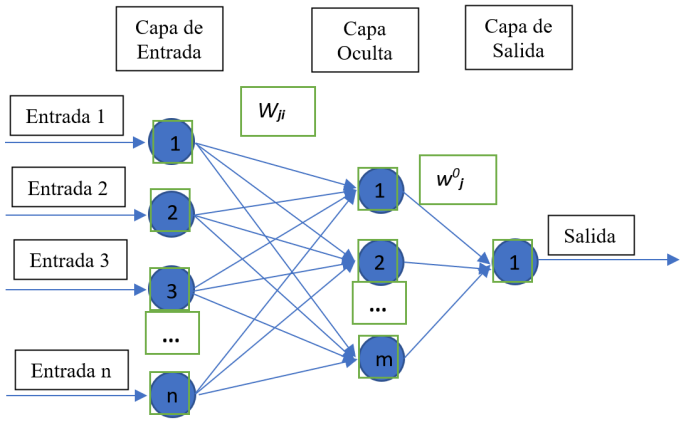

Figura 1 Estructura de una red neuronal.

2. Métodos

Una neurona recibe un conjunto de señales a través de las dendritas que son las vias de comunicación para ellas, dicha información representa los datos del estado de activación de todas las neuronas con las que se encuentra conectadas. Sea x_i el valor de entrada de la neurona en un instante y la conexión entre la neurona está ponderado por un peso w_{ji}. Entonces considerando un efecto aditivo, la entrada neta que recibe una neurona está dada por ecuación 1.

$$f_j(x) = w_0 + \sum_{i=1}^{N} w_{ji} x_i \qquad (1)$$

La función que realiza una red neuronal depende de la función de activación y de las conexiones entre nodos, cuando alguno de los nodos recolecta la información necesaria, rápidamente envía los datos adquiridos a través de vías de comunicación inalámbrica hacia un punto donde se administran todos los datos recolectados por cada nodo. La función de activación es en parte responsable de la bondad del comportamiento de la red neuronal. La salida de una neurona se denota por y, y se obtiene mediante la aplicación de la función de transferencia. Entre las funciones de activación o transferencia más usadas tenemos: Escalón, logística, tangente hiperbólica, secante e identidad. Al procedimiento mediante el que se modifica el valor de los pesos de conexión para obtener la salida deseada se le denomina aprendizaje o entrenamiento. En la red neuronal se usa el concepto de función de error como una medición de la discrepancia entre lo predicho por la red neuronal y

lo observado, lo que es equivalente al error cuadrático medio. En la red neuronal, lo que se intenta, en general, es minimizar la suma del error cuadrático, ecuación 2.

$$Error(p) = \frac{1}{2}\sum_{j=1}^{M}(predicción_j(p) - observación_j(p))^2 \quad (2)$$

Para cada iteración p de cálculo en la red, o también llamadas épocas, el error es el indicador para actualizar los pesos de cada conexión a través de la regla de aprendizaje, que en el aprendizaje supervisado es muy común usar el algoritmo del gradiente descendente, ecuación 3.

$$\Delta w_{ji}(p) = -\eta \frac{\partial Error(p)}{\partial v_j(p)} \quad (3)$$

Donde v_j es el campo local inducido sobre la cual se calcula la regla de aprendizaje, y_i es la salida de la neurona anterior, y η es un factor de aprendizaje, cuyo valor generalmente se encuentra entre 0.2 y 0.8. Desde un punto de vista muy global se distinguen dos tipos de aprendizaje: supervisado y no supervisado. El aprendizaje supervisado trata de conseguir que la red sea capaz de predecir, a partir de un conjunto de características suministradas como entradas, el valor que tomarán otras características, llamadas objetivo, habiendo sido observadas ambos tipos de características en cada uno de los patrones de entrenamiento. El aprendizaje se lleva a cabo modificando los pesos de la red según alguna regla que trata de minimizar la discrepancia entre la salida de la red y la salida correcta proporcionada por los objetivos. El proceso de aprendizaje comienza una vez definida la arquitectura de la red. A los datos utilizados en el proceso de aprendizaje se les denomina conjunto de entrenamiento. Durante el aprendizaje se procesan varias veces los pares de vectores del conjunto de entrenamiento y en cada paso del aprendizaje se modifican los pesos de las conexiones. Terminado el proceso de aprendizaje, la red se evalúa sobre un conjunto de datos no utilizados en el entrenamiento y denominado conjunto de validación. En el aprendizaje no supervisado existen características de entrada, pero no objetivo. No existe información que indique si la salida que produce la red es o no correcta. Las técnicas de aprendizaje no supervisado se utilizan sobre todo para obtener estructuras,

relaciones o clasificaciones. Existen muchos algoritmos de aprendizaje, pero los más extendidos son la regla delta generalizada o backpropagation para el aprendizaje supervisado, y la regla estándar del aprendizaje competitivo en el caso no supervisado. En el trabajo de Menacho (2014) se implementa un modelo de regresión lineal con redes neuronales, ver [7] para más información.

3. Descripción de los datos

Las posibilidades de aplicación de las redes neuronales a problemas de investigación abordados tradicionalmente desde las técnicas estadísticas clásicas son múltiples. El problema que se aborda en este artículo, con la implementación de una RNA en Excel es el ajuste por regresión lineal múltiple del rendimiento de cosecha del Brócoli en una granja agrícola local en Aguascalientes, México. Se supone, además, la dependencia de esta serie de tiempo real con seis variables meteorológicas independientes también reales: Precipitación, Temperatura mínima, Temperatura base óptima, Temperatura Vernalización/Desvernalización, Temperatura cero de crecimiento y Temperatura mínima/máxima. El muestreo se realizó durante el año 2019, con 2100 observaciones. En este ejemplo se programó una MACRO de Excel® para una red neuronal de seis entradas, seis neuronas en la capa oculta y una salida. La arquitectura de esta RNA se muestra a detalle en a figura 2.

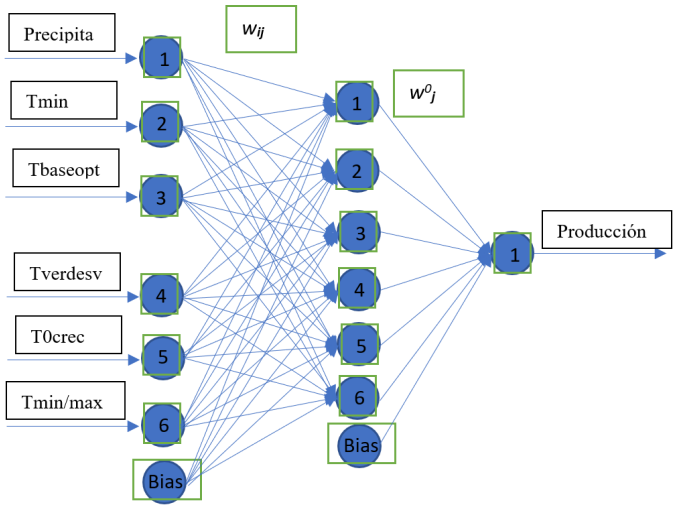

Figura 2 Estructura de la red neuronal 6-6-1.

Entonces se requiere desarrollar un análisis de Regresión Lineal Múltiple con un 99% de confianza. Para realizarlo, en lugar de usar software especializado como MINTAB, SPSS; STATISTICA, o NCSS; o bien algún lenguaje de programación, sólo utilizaremos una hoja de cálculo con una MACRO programada en EXCEL®. Para eso, debemos contar con los datos disponibles en la hoja de cálculo y dividir el total de observaciones en datos de entrenamiento y datos de prueba, en una proporción aproximada de 80 y 20%, respectivamente.

En el Rancho Medio Kilo, el brócoli convencional representa el 39% de la producción total de la finca. En periodos de cosecha la finca debe contratar personal, y debe preparar maquinaria, equipo y recursos adicionales. Contar con estimaciones de la producción total anual del brócoli, le permite a la finca tomar decisiones en cuanto a la planificación estacional del cultivo, y para equilibrar la cantidad de medios de producción utilizados, como mano de obra humana, maquinaria, anticipación de recursos, y también para estar preparados para reaccionar contra plagas o condiciones climáticas inesperadas. El rendimiento del cultivo se mide en kilogramos por hectárea (kg/Ha). La evaluación del pronóstico del rendimiento del cultivo se mide con respecto a su valor real en el siguiente intervalo de variación de [-15%, 15%]. El modelo estadístico utilizado para describir la producción total del brócoli en este trabajo es un modelo lineal múltiple de regresión (MLR) utilizando una RNA e implementada en una hoja de cálculo de Excel® con el uso de MACROS. Los datos meteorológicos se recolectaron en la finca de mayo a diciembre de 2019. Los datos del clima fueron generados por una de las estaciones meteorológicas operadas por la oficina de agricultura mexicana llamada Instituto Nacional de Investigaciones Forestales Agrícolas y Pecuarias (INIFAP), en Aguascalientes, México. Los datos meteorológicos tienen nueve variables que incluyen precipitación, radiación y temperaturas, como mínimo-máximo, base óptima, vernalización-desvernalización, crecimiento cero y crecimiento óptimo. La tabla 1 presenta las abreviaturas utilizadas aquí para las variables meteorológicas y la figura 3 ilustra el gráfico de estos predictores. Partiremos del comportamiento de las variables, mostrado en la figura 3, que primero incluye a las variables predictoras meteorológicas y posteriormente incluye a la producción total. Note que la escala

en esta última variable es mucho mayor que la escala de las variables predictoras y por eso se muestra en una gráfica aparte.

Tabla 1 Predictores meteorológicos para la producción total del brócoli.

Variable	Descripción
Precipitation	Precipitación
Radiation	Radiación solar
Tmin	Temperatura mínima
Tmax	Temperatura máxima
Tbaseopt	Temperatura base óptima
Tverdesv	Temperatura Vernalización-Desvernalización
T0grow	Temperatura cero de crecimiento
Tminmax	Temperatura Mínima-Máxima
Toptgrow	Temperatura de crecimiento óptima

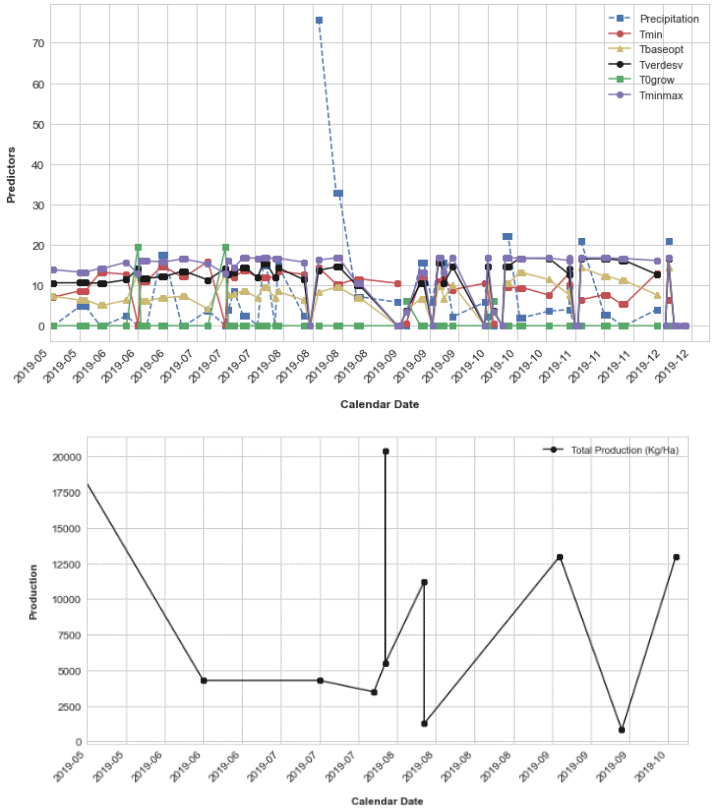

Figura 3 Datos de entrenamiento de la red neuronal.

4. Resultados

En este apartado describiremos la programación de la MACRO en Visual Basic y la secuencia de operaciones realizadas en Excel® para implementar la red neuronal de la figura 2.

El algoritmo de la RNA puede dividirse en tres etapas de implementación, el entrenamiento, la etapa de prueba y la validación. En la etapa de entrenamiento, el objetivo del algoritmo es definir los valores de los pesos que mejor estimen la producción real disponible en los datos. El diagrama de flujo de la figura 4 ilustra el funcionamiento del entrenamiento de la RNA para cumplir con el objetivo anterior.

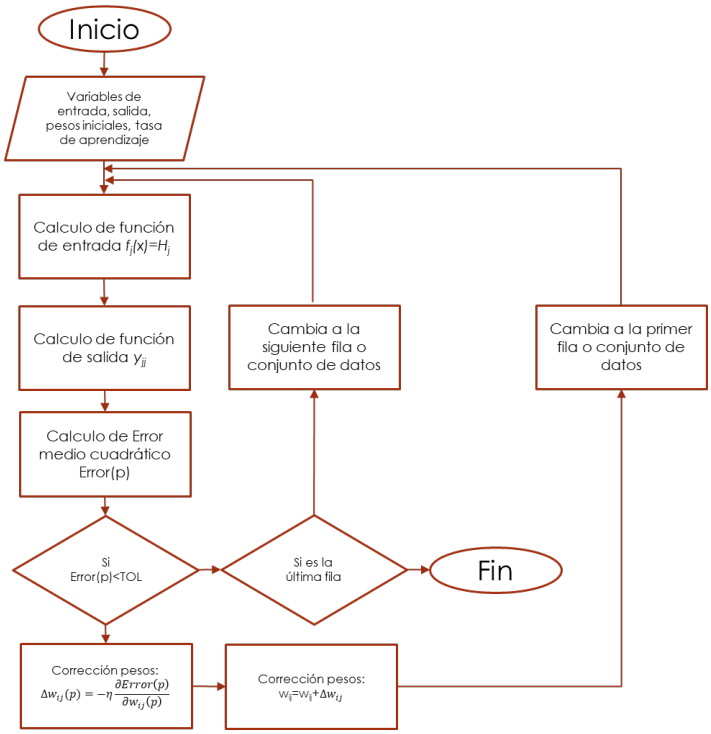

Figura 4 Diagrama de flujo para la etapa de entrenamiento de la RNA.

Para crear una MACRO de Excel® debe activarse el entorno de programación. En el entorno de programación, el primer paso es declarar la instrucción para indicar el registro en el cual debe de posicionarse el cursor para comenzar a ejecutar el entrenamiento. Posteriormente, se agrega el ciclo principal, cuya finalidad es gobernar las iteraciones globales del entrenamiento. Todas las acciones dentro de él se estarán ejecutando, para todos los registros de la tabla. Y sólo se detendrá hasta que llegue a un registro o fila de datos vacía. Una vez declarado el ciclo principal, ahora se declaran las variables correspondientes a los pesos aleatorios iniciales, así como también la tasa de aprendizaje, cuyo valor permanecerá constante durante todo el ciclo de entrenamiento de la RNA, figura 5.

```
Range("A2").Select

Do While Range("B" & ActiveCell.Row).Value <> Empty


w1 = 0.023
w2 = 0.014
w3 = 0.015
w4 = 0.013
w5 = 0.014
w6 = 0.015
w7 = 0.023
w8 = 0.024
w9 = 0.023
w10 = 0.028
w11 = 0.037
w12 = 0.024
w13 = 0.012
w14 = 0.01
w15 = 0.013
w16 = 0.019
w17 = 0.018
w18 = 0.017
w19 = 0.016
w20 = 0.013
w21 = 0.026
w22 = 0.023
w23 = 0.028
w24 = 0.027
w25 = 0.024
w26 = 0.027
w27 = 0.054
w28 = 0.034
w29 = 0.017
w30 = 0.017
w31 = 0.036
w32 = 0.029
w33 = 0.018
w34 = 0.027
w35 = 0.033
w36 = 0.017
w37 = 0.023
w38 = 0.031
w39 = 0.011
w40 = 0.012
w41 = 0.027
w42 = 0.028
w43 = 0.049
w44 = 0.035
w45 = 0.044
w46 = 0.019
w47 = 0.027
w48 = 0.038
w49 = 0.035
Bias = 1


TA = 0.00000000001
```

Figura 5 Inicio de programa para la MACRO.

Los valores numéricos iniciales de los pesos w_{ji} y w_j^0 se especifican de manera manual. El único requisito para estos valores es que sean valores aleatorios entre 0 y 1. Posteriormente, cada valor numérico de los pesos se actualiza con la regla de aprendizaje de la ecuación 3, que consiste en la diferencia entre el valor del peso actual y corrección.

Una vez que se definieron los valores iniciales de los pesos, tasa de aprendizaje y sesgo, lo siguiente es indicar al programa la ubicación de la tasa de aprendizaje. Se incluye a continuación una instrucción para definir un ciclo interno, cuya función es el de calcular la función de entrada de la capa oculta de la RNA. Y se ejecutan todas las instrucciones dentro de él, hasta que se cumpla la condición de paro. Después se indica al programa la ubicación de los primeros pesos aleatorios de la capa de

entrada, dentro de la hoja de Excel. Con estos pesos se calcula la función de entrada, cuyos valores constituyen los resultados de las neuronas de la capa oculta, figura 6.

```
Range("BQ2").Value = TA
Do While Range("J" & ActiveCell.Row).Value <= Range("I" & ActiveCell.Row).Value

 Range("K" & ActiveCell.Row).Value = w1
 Range("L" & ActiveCell.Row).Value = w2
 Range("M" & ActiveCell.Row).Value = w3
 Range("N" & ActiveCell.Row).Value = w4
 Range("O" & ActiveCell.Row).Value = w5
 Range("P" & ActiveCell.Row).Value = w6
 Range("Q" & ActiveCell.Row).Value = w7
 Range("R" & ActiveCell.Row).Value = w8
 Range("S" & ActiveCell.Row).Value = w9
 Range("T" & ActiveCell.Row).Value = w10
 Range("U" & ActiveCell.Row).Value = w11
 Range("V" & ActiveCell.Row).Value = w12
 Range("W" & ActiveCell.Row).Value = w13
 Range("X" & ActiveCell.Row).Value = w14
 Range("Y" & ActiveCell.Row).Value = w15
 Range("Z" & ActiveCell.Row).Value = w16
 Range("AA" & ActiveCell.Row).Value = w17
 Range("AB" & ActiveCell.Row).Value = w18
 Range("AC" & ActiveCell.Row).Value = w19
 Range("AD" & ActiveCell.Row).Value = w20
 Range("AE" & ActiveCell.Row).Value = w21
 Range("AF" & ActiveCell.Row).Value = w22
 Range("AG" & ActiveCell.Row).Value = w23
 Range("AH" & ActiveCell.Row).Value = w24
 Range("AI" & ActiveCell.Row).Value = w25
 Range("AJ" & ActiveCell.Row).Value = w26
 Range("AK" & ActiveCell.Row).Value = w27
 Range("AL" & ActiveCell.Row).Value = w28
 Range("AM" & ActiveCell.Row).Value = w29
 Range("AN" & ActiveCell.Row).Value = w30
 Range("AO" & ActiveCell.Row).Value = w31
 Range("AP" & ActiveCell.Row).Value = w32
 Range("AQ" & ActiveCell.Row).Value = w33
 Range("AR" & ActiveCell.Row).Value = w34
 Range("AS" & ActiveCell.Row).Value = w35
 Range("AT" & ActiveCell.Row).Value = w36
 Range("AU" & ActiveCell.Row).Value = w37
 Range("AV" & ActiveCell.Row).Value = w38
 Range("AW" & ActiveCell.Row).Value = w39
 Range("AX" & ActiveCell.Row).Value = w40
```

Figura 6 Código para insertar valores de los pesos y definición de ciclo interno.

Las figuras 7 y 8 muestran las variables de entrada x_j y pesos numéricos iniciales para cada conexión en las neuronas de la entrada w_{ji}. Los pesos de la capa oculta w_j^0 serán asignados posteriormente. La neurona cero representa al sesgo en la RNA y su valor es un valor constante uno. En total, son 42 pesos que permiten la conexión desde la capa de entrada hasta la capa oculta, seis conexiones por cada neurona de entrada, ya que la neurona del sesgo en la capa oculta no requiere de una conexión con las variables de entrada. En total, son siete pesos que permiten la conexión desde la capa oculta hasta la capa de salida, figura 2.

BH451 | 0.0490021093899848

	A	B	C	D	E	F	G	H	I
1	Fecha	X0	PRECIP (X1)	T MIN (X2)	T BASE OPT (X3)	T VER/DESV (X4)	T MIN/MAX (X5)	Escenario 3 (X6)	Produccion Total (Kg/Ha) (Y)
2	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
3	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
4	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
5	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
6	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
7	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37
8	03/12/2019	1	0	3.3	118.25	154.5	146	1560.25805	4295.37

Figura 7 Variables de entrada de la capa de entrada de la neurona.

J2 | 4294.47266427534

	K	L	M	N	O	P	Q	R	S	T
1	Peso (W1)	Peso (W2)	Peso (W3)	Peso (W4)	Peso (W5)	Peso (W6)	Peso (W7)	Peso (W8)	Peso (W9)	Peso (W10)
2	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505
3	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505
4	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505
5	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505
6	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505
7	0.029122551	0.014	0.035204419	0.736991673	0.959934152	0.908892467	9.575759714	0.027738638	0.023	0.040337505

Figura 8 Pesos numéricos iniciales de cada neurona en la capa de entrada.

Cuando llega a la capa oculta, la información de entrada, se calcula la función de entrada utilizando la ecuación 1. El valor numérico de esta función de entrada se asigna a la nueva variable H_i. En total son seis valores de esta función de entrada, desde la neurona uno a la neurona seis en la capa oculta. La neurona asignada al sesgo en la capa oculta también conserva un valor constante igual a uno. Ver la arquitectura de la RNA en la figura 2. En la hoja de cálculo, esto se calcula mediante el siguiente código.

En el código de la figura 9, se declara el proceso para calcular cada una de las seis funciones de entrada de la capa oculta, así como también la ubicación donde se mostrarán en la hoja de Excel. Posteriormente, se agrega en el código la ubicación de la variable de sesgo en esa misma capa, ya que esta formara parte de las neuronas de la capa oculta. Es decir, la capa oculta tiene un numero de siete neuronas que la forman. Después, se le indica al programa que deberá agregar los valores de los últimos siete pesos restantes dentro de la hoja de Excel. Estos pesos influyen en los cálculos realizados entre la capa oculta y la capa de salida, figura 10. En la figura 10 se muestran las columnas de los pesos de la capa de salida. Recuerde que los valores numéricos iniciales de los pesos w^0_j también se generan de manera aleatoria entre 0 y 1. La figura 11 muestra los valores numéricos generados en cada función de entrada H_i, en la capa oculta.

```
Range("BA" & ActiveCell.Row).Value = Bias
Range("BB" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("K" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("L" & ActiveCell.Row).Value) +
Range("BC" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("R" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("S" & ActiveCell.Row).Value) +
Range("BD" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("Y" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("Z" & ActiveCell.Row).Value) +
Range("BE" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("AF" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("AG" & ActiveCell.Row).Value)
Range("BF" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("AM" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("AN" & ActiveCell.Row).Value)
Range("BG" & ActiveCell.Row).Value = (Range("B" & ActiveCell.Row).Value * Range("AT" & ActiveCell.Row).Value) + (Range("C" & ActiveCell.Row).Value * Range("AU" & ActiveCell.Row).Value)


Range("BH" & ActiveCell.Row).Value = w43
Range("BI" & ActiveCell.Row).Value = w44
Range("BJ" & ActiveCell.Row).Value = w45
Range("BK" & ActiveCell.Row).Value = w46
Range("BL" & ActiveCell.Row).Value = w47
Range("BM" & ActiveCell.Row).Value = w48
Range("BN" & ActiveCell.Row).Value = w49
```

Figura 9 Declaración de código de cálculo de las funciones de entrada de la capa oculta.

J2 | 4294.47266427534

	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN
1	H5	H6	H7	Peso (W43)	Peso (W44)	Peso (W45)	Peso (W46)	Peso (W47)	Peso (W48)	Peso (W49)
2	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
3	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
4	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
5	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
6	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
7	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257
8	24895.44425	22598.91558	18621.28157	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257

Figura 10 Pesos numéricos iniciales para cada neurona en la capa oculta.

J2 | 4294.47266427534

	AZ	BA	BB	BC	BD	BE	BF	BG	BH
1	Peso (W42)	H1	H2	H3	H4	H5	H6	H7	Peso (W43)
2	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
3	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
4	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
5	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
6	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
7	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473
8	11.64764245	1	15308.95887	9348.17097	17200.47918	24895.44425	22598.91558	18621.28157	0.049002473

Figura 11 Funciones de entrada de cada neurona en la capa oculta, incluyendo el sesgo.

El valor numérico de cada función de entrada, en la capa oculta, así como el sesgo en la misma capa, se utilizan para generar la estimación de la salida y_p, con las respectivas conexiones y sus pesos numéricos, etiquetados como w_j^0. Desde el punto de vista práctico, la salida y_p es una nueva función de entrada para la capa de salida, donde sus entradas son las funciones H_i. En Excel®, este cálculo se realiza con el siguiente código. Se agrega el código para obtener el valor de predicción de la salida o producción total, de la neurona de salida. Con dicha predicción, se obtiene el valor del error cuadrático medio y el valor de la diferencia entre la producción real y la predicción anterior, definida por la variable Δ, figura 12.

```
Range("J" & ActiveCell.Row).Value = (Range("BA" & ActiveCell.Row).Value * Range("BH" & ActiveCell.Row).Value) + (Range("BB" & ActiveCell.Row).Value
Range("BO" & ActiveCell.Row).Value = 0.5 * ((Range("J" & ActiveCell.Row).Value - Range("I" & ActiveCell.Row).Value) ^ 2)
Range("BP" & ActiveCell.Row).Value = (Range("J" & ActiveCell.Row).Value - Range("I" & ActiveCell.Row).Value)
```

Figura 12 Declaración de código de predicción, error medio cuadrático y valor de Δ.

Con el error medio cuadrático entre el valor estimado y el valor real de la producción total, utilizando la ecuación 2, es posible definir una condición de paro para el ciclo principal en la MACRO de Excel®. Cuando este error es menor a cierta tolerancia establecida, por ejemplo, $TOL = 1 \times 10^{-3}$, significa que la estimación obtenida de la RNA ha sido precisa y entonces, o se continua con la siguiente fila o conjunto de datos, o se termina el algoritmo de entrenamiento de la red. En el caso en que la producción estimada difiera significativamente de la producción real, es decir que el error medio cuadrático sea mayor a la tolerancia establecida, el algoritmo de entrenamiento de la RNA debe corregir los pesos actuales con la regla de aprendizaje analizada en ecuación 2, en propagación hacia atrás, es decir, primero se actualizan los pesos de la capa de salida, y posteriormente se actualizan los pesos de la capa oculta. En ambos casos, el factor de aprendizaje será un valor constante entre cero y uno. En la hoja de cálculo, este valor aparece en la celda BQ2 y es igual a $\eta = 1E^{-11}$. El campo local inducido sobre la cual se calcula la regla de aprendizaje, v_j, se elige igual a los pesos de la capa de entrada w_{ji}, o de la capa oculta w_j^0, respectivamente. Esto es claro cuando se entiende que, para reducir el error de estimación en el algoritmo, los únicos parámetros de cambio son los pesos propuestos.

Se utiliza la regla de la cadena para el cálculo de la regla de aprendizaje para incluir la variable estimada $predicciónj(p)$, y así obtener una forma numérica y práctica de calcularla. En la propagación hacia atrás, primero se calcula la $\Delta w_j^0(p) = -\eta \frac{\partial Error(p)}{\partial w_j^0(p)}$, obteniendo ecuación 4.

$$\Delta w_j^0(p) = -\eta(predicción_j(p) - observación_j(p))H_i(p) \qquad (4)$$

La regla de aprendizaje para las neuronas de la capa oculta se calcula con una ligera variación de la regla de aprendizaje anterior. En este caso se utiliza dos veces la regla de la cadena para el cálculo de la regla de aprendizaje con el propósito de

incluir tanto a la variable estimada $predicciónj(p)$, como a la variable H_i, y así obtener una forma numérica práctica de calcularla. En la propagación hacia atrás, se calcula entonces la $\Delta w_{ij}(p) = -\eta \frac{\partial Error(p)}{\partial w_{ij}(p)}$, y se obtiene la regla de aprendizaje, ecuación 5.

$$\Delta w_{ij}(p) = -\eta(predicción_j(p) - observación_j(p))entrada_i(p)w_j^0(p) \qquad (5)$$

En términos de la MACRO de Excel®, las reglas de aprendizaje obtenidas en las ecuaciones 4 y 5, se calculan de la siguiente manera. El programa es capaz de aprender por sí mismo con la regla de aprendizaje y actualizar los pesos correctos que estimen la producción total. Como primer paso, se declaran las operaciones matemáticas para recalcular los siete últimos pesos, entre la capa oculta y la capa de salida, figura 13. La figura 14 muestra los valores numéricos generados con el código anterior.

```
w49 = Range("BN" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w48 = Range("BM" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w47 = Range("BL" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w46 = Range("BK" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w45 = Range("BJ" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w44 = Range("BI" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w43 = Range("BH" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("BP" & ActiveCell.Row).Value * Range("BA" & ActiveCell.Row).Value)
```

Figura 13 Código de autoentrenamiento de los pesos entre la capa oculta y la de salida.

BQ2 0.00000000001

	BH	BI	BJ	BK	BL	BM	BN	BO	BP	BQ
1	Peso (W43)	Peso (W44)	Peso (W45)	Peso (W46)	Peso (W47)	Peso (W48)	Peso (W49)	Error = $\frac{1}{2}$(prediction − actual)^2	Δ = prediction − actual	a
2	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	1E-11
3	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	
4	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	
5	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	
6	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	
7	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	
8	57	0.049002473	0.041122551	0.047738638	0.025879032	0.036956499	0.047038042	0.042447257	0.402605701	-0.897335725	

Figura 14 Regla de aprendizaje para los pesos numéricos entre la capa oculta y la de salida (Backpropagation).

Una vez recalculados los pesos anteriores, se genera el código para recalcular los primeros 42 pesos entre la capa de entrada y la capa oculta, figura 15.

En la figura 13 se muestran algunos de los valores numéricos obtenidos del código anterior. Cuando los pesos se actualizan con la corrección establecida por la regla de aprendizaje, con propagación hacia atrás, el algoritmo vuelve a calcular todas las estimaciones de la producción desde la fila o conjunto inicial de valores de entrada. A cada una de estas iteraciones se les denomina épocas y se etiquetan

con el subíndice *p*. El algoritmo de entrenamiento de la RNA termina cuando se calculan los pesos de la última fila o conjunto de valores en la base de datos.

```
w42 = Range("AZ" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w41 = Range("AY" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w40 = Range("AX" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w39 = Range("AW" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w38 = Range("AV" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w37 = Range("AU" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w36 = Range("AT" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BG" & ActiveCell.Row).Value)
w35 = Range("AS" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w34 = Range("AR" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w33 = Range("AQ" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w32 = Range("AP" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w31 = Range("AO" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w30 = Range("AN" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w29 = Range("AM" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BF" & ActiveCell.Row).Value)
w28 = Range("AL" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w27 = Range("AK" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w26 = Range("AJ" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w25 = Range("AI" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w24 = Range("AH" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w23 = Range("AG" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w22 = Range("AF" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BE" & ActiveCell.Row).Value)
w21 = Range("AE" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w20 = Range("AD" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w19 = Range("AC" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w18 = Range("AB" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w17 = Range("AA" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w16 = Range("Z" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w15 = Range("Y" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BD" & ActiveCell.Row).Value)
w14 = Range("X" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w13 = Range("W" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w12 = Range("V" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w11 = Range("U" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w10 = Range("T" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w9 = Range("S" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w8 = Range("R" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BC" & ActiveCell.Row).Value)
w7 = Range("Q" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("H" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w6 = Range("P" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("G" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w5 = Range("O" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("F" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w4 = Range("N" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("E" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w3 = Range("M" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("D" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w2 = Range("L" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("C" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
w1 = Range("K" & ActiveCell.Row).Value - (Range("BQ2").Value * Range("B" & ActiveCell.Row).Value * Range("BP" & ActiveCell.Row).Value * Range("BB" & ActiveCell.Row).Value)
```

Figura 15 Código para recalcular los primeros 42 pesos.

Después de declarar las instrucciones para recalcular todos los pesos de la RNA, se agrega un ciclo condicional para evaluar si la predicción cumple con la condición de paro, respecto al valor del error medio cuadrático y la tolerancia establecida. Si la condición es verdadera, el programa entiende que saldrá del primer ciclo y seguirá al próximo, y el ciclo volverá a empezar. Si la condición es falsa, entonces comenzará la actualización de los pesos y volverá a reiniciar el ciclo principal con los nuevos valores de los pesos corregidos. Esto lo hará sucesivamente hasta que ya no queden registros para terminar con el programa, figura 16.

```
If Range("J" & ActiveCell.Row).Value > Range("I" & ActiveCell.Row).Value - 1 Then
 Exit Do
Else
  Range(("J" & ActiveCell.Row), ("BP" & ActiveCell.Row)).Clear
End If


Loop

Range("A" & ActiveCell.Row + 1).Select


Loop

End Sub
```

Figura 16 Código con instrucción condicional de paro.

Una vez implementado el algoritmo anteriormente mencionado se observo que el programa comenzó a arrojar datos satisfactorios ya que los pesos aleatorios comenzaron a afinarse y a reducir su margen de cambios hasta que los cambios fueron demasiado pequeños y que ya era difícil notar algún cambio.

5. Conclusiones

Si bien las herramientas del aprendizaje automático, como parte de la inteligencia artificial, ya tienen muchos años de existencia, el gran avance tecnológico en computadoras permite resolver numéricamente diversos problemas de manera rápida y sencilla. Sin embargo, el uso de esos algoritmos sólo está disponible en software especializado o en lenguajes de programación. Eso hace que el acceso y la apropiación social de este tipo de algoritmos no esté al alcance de cualquier persona. En este sentido, la traducción de las herramientas del aprendizaje automático a herramientas de cálculo común como Excel, permite que más personas apliquen este tipo de algoritmos a la resolución de problemas en su quehacer diario. En este artículo se revisa paso a paso la implementación de una red neuronal para resolver un problema de regresión en una hoja de Excel® con el uso de MACROS. Las fórmulas y resultados obtenidos también se muestran de manera explícita para que puedan ser replicados por cualquier persona con conocimientos básicos de Excel®.

De igual manera el uso de redes neuronales en Excel® para el análisis de datos agrícolas representa una herramienta prometedora para la toma de decisiones en la agricultura ya que con estas redes se pueden descubrir patrones complejos y relaciones en los datos que son cruciales para optimizar la gestión de cultivos, predecir cosechas. Sin embargo, es esencial destacar la importancia de contar con datos de alta calidad, recursos computacionales adecuados y una comprensión sólida de las redes neuronales y su interpretación. Además, las redes neuronales deben considerarse como una parte complementaria en el análisis agrícola, junto con enfoques tradicionales y experiencia práctica en la agricultura, para maximizar su utilidad y garantizar decisiones informadas y confiables en este campo crítico.

6. Bibliografía y Referencias

[1] Segee, B. E., & Amos, M. D. (1997, June). Artificial Neural Networks Using Microsoft Excel For Windows 95. In 1997 Annual Conference (pp. 2-81).

[2] Thin-Yin, L., & Jonathan, L. Y. (2020). Simpler Machine Learning Using Spreadsheets: Neural Network Predict. European Journal of Formal Sciences and Engineering, 3(2), 43-64.

[3] García, J. (2002). Hojas de cálculo para la simulación de redes de neuronas artificiales (RNA). Qüestiió: quaderns d'estadística i investigació operativa, 289-305.

[4] Semerikov, S., Teplytskyi, I., Yechkalo, Y., Markova, O., Soloviev, V., & Kiv, A. (2020). Using spreadsheets as learning tools for computer simulation of neural networks. In SHS Web of Conferences (Vol. 75, p. 04018). EDP Sciences.

[5] Segee, B. E., & Amos, M. D. (1997, June). Artificial Neural Networks Using Microsoft Excel For Windows 95. In 1997 Annual Conference (pp. 2-81).

[6] Demir, S., DEMİR, N. M., Karadeniz, A., & YÖRÜKLÜ, H. C. (2018). Implementation of an MS Excel tool for backpropagation neural network algorithm in environmental engineering education. Sigma Journal of Engineering and Natural Sciences, 36(1), 251-260.

[7] Menacho Chiok, C. H. (2014). Modelos de regresión lineal con redes neuronales.