TAX PREDICTION ARTIFICIAL NEURAL NETWORK USING SIGMOID FUNCTION AS THE ACTIVATION FUNCTION

Charles Ndung'u

April 2023

WWW.KNCMAP.COM

Contents

1 Input Layer

The first thing when designing a neural network is to have a good volume of data, thus we shall begin by loading the data. For simplicity of the Neural Network model, we shall use Python as our programming language, and the preferred computing servers are Colab, which are free cloud computing servers for machine learning.

Neural Network Architecture:

We shall use the following architecture for our Neural Network:

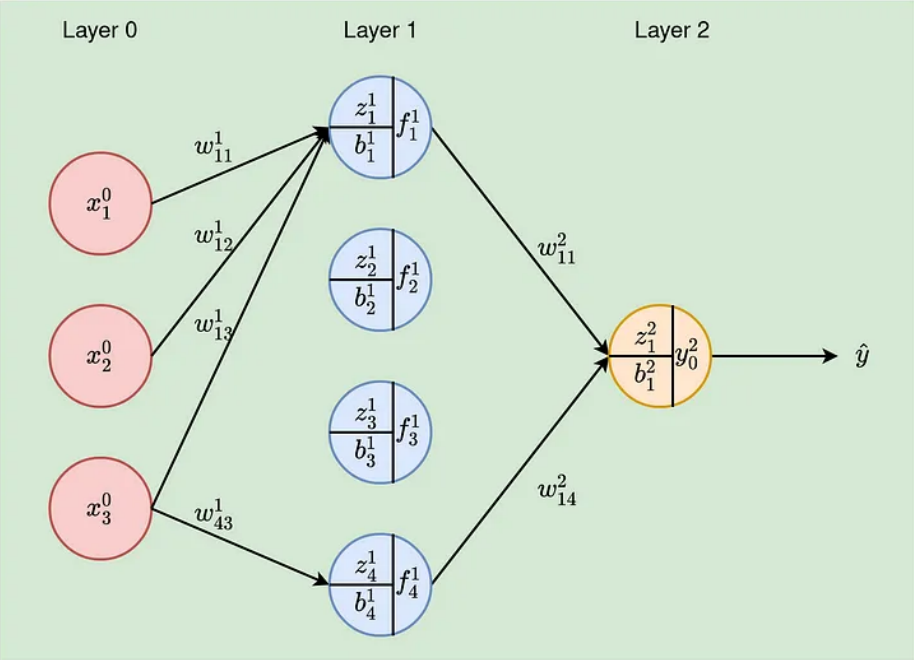
Input layer (1 neuron) → Hidden layer (5 neurons) → Output layer (1 neuron)

Definitions:

- **Neuron:** A neuron typically receives one or more inputs, performs a computation on these inputs, and produces an output that can be passed on to other neurons in the network.
- **Activation function:** An activation function is a mathematical function that is applied to the output of each neuron in a neural network.

- **Weights:** Weights refer to the parameters that are associated with the connections between neurons. Each connection between neurons has an associated weight that determines the strength of the connection.
- **Bias:** The bias is a scalar value that is added to the output of a neuron before applying the activation function.

Diagram representation of a neural network:



Analysis of the Diagram/Neural Network Architecture:

- x_1^0, x_2^0, x_3^0 represent the input values. For example, x_1^0 shows the first input in layer 0.
- $W_{1_1}^0$ represents the weight coming from layer 0 to the neuron x_1^0.

Suppose we have a neural network with one input neuron and one hidden neuron. The input neuron receives a scalar input value x, and the hidden neuron receives the output of the input neuron as input, with a scalar weight w. The output z of the hidden neuron is calculated as:

$$z = f(w \cdot x)$$

where f is the activation function (in this case, the sigmoid function):

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

During training, the weight w is adjusted to minimize the error between the output and the expected output for a given input x. A common method for weight adjustment is `backpropagation`. Once trained, the weight w is fixed and can be used to make predictions on new input data.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

2 Neural Network Layer

First, let's define the input and output of our model:

- Input: Current tax
- Output: Estimated tax

Now, let's define the architecture of our neural network. For this example, we're using a simple feedforward neural network with one hidden layer. The number of neurons in the input layer is 1, and the number of neurons in the output layer is also 1. The number of neurons in the hidden layer can be adjusted to improve the model's performance.

The architecture is as follows:

Input layer (1 neuron) $\rightarrow$ Hidden layer (5 neurons) $\rightarrow$ Output layer (1 neuron)

Now, let's define the activation function for each neuron. Since we are using the sigmoid activation function, the equation is:

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

This function will map inputs and weights to values between 0 and 1.

We can train the model using a dataset of current taxes and their corresponding estimated taxes. Once the model is trained, we can use it to estimate taxes for new inputs.

```
from keras.models import Sequential
from keras.layers import Dense
import numpy as np

# Define the neural network architecture
model = Sequential()
model.add(Dense(5, input_dim=1, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(loss='mean_squared_error', optimizer='adam')

# Define the training data
X_train = np.array([1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0])
y_train = np.array([1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9, 11.0])

# Train the model
model.fit(X_train, y_train, epochs=1000, batch_size=10)

# Use the model to estimate taxes for a new input
X_new = np.array([11.0])
y_new = model.predict(X_new)
print(y_new)
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

3 Output Layer

Here's how the program works, step by step:

1. Import the necessary libraries (e.g., Keras) for building and training the neural network.
2. Define the architecture of our neural network. The input layer has one neuron, the hidden layer has five neurons, and the output layer has one neuron.
3. Compile the model by specifying the loss function and optimizer. We use mean squared error as the loss function and the Adam optimizer.
4. Define the training data, which consists of current taxes and their corresponding estimated taxes.
5. Train the model using the `fit` method. The model is trained for 1000 epochs with a batch size of 10.
6. Finally, use the model to estimate taxes for a new input of 11.0.

The code defines a neural network with one input neuron, a hidden layer with five neurons, and one output neuron, using the sigmoid activation function for each neuron.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.