DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GOVERNMENT ENGINEERING COLLEGE THRISSUR

MACHINE LEARNING ASSIGNMENT SEMESTER 7

Submitted By: Marrea Treasa Francis

Reg No: TCR22CS046

TITLE

Sequence Summation using Feedforward Neural Network

OBJECTIVE

To design and train a feedforward neural network that learns to compute the sum of a sequence of input numbers using supervised learning, enabling the model to generalize summation behavior to unseen input sequences.

ALGORITHM

1. **Initialize Parameters:**
   Randomly assign small weights and biases within the range [-0.5, 0.5]. Set learning rate η = 0.1.
2. **Forward Propagation:**
   - Input layer receives three input values (sequence).
   - Hidden layer (4 neurons) computes activations using the Sigmoid function.
   - Output layer (1 neuron) applies a Linear activation to produce the predicted sum.
3. **Error Calculation:**
   Compute the mean squared error (MSE) between predicted and actual sums.
4. **Backward Propagation:**
   - Calculate gradients of the error with respect to each weight.
   - Update weights using:
     $$w_{new} = w_{old} + \eta \times \text{error} \times \text{input}$$
5. **Iteration:**
   Repeat steps 2–4 across all training samples for 800 epochs or until error converges below the desired threshold.
6. **Testing:**
   Evaluate the trained network on unseen sequences and compare predicted outputs with actual sums to verify accuracy and generalization.

NETWORK STRUCTURE

Layer	No. of Neurons	Activation Function
Input Layer	3	—
Hidden Layer	4	Sigmoid
Output Layer	1	Linear

TRAINING PARAMETERS

- **Learning Rate:** 0.1
- **Epochs:** 800
- **Error Function:** Mean Squared Error (MSE)

CHANGES MADE

- Increased hidden layer neuron count from **3 → 4** for improved learning capacity.
- Reduced **learning rate from 0.5 → 0.1** to ensure smoother convergence.
- Implemented **random weight initialization** in the range **[-0.5, 0.5]** to prevent neuron saturation.
- Modified training dataset to represent **sequence summation patterns** instead of fixed rules.
- Added **real-time training error tracking and print statements** for progress visualization.

SAMPLE INPUT AND OUTPUT

Test Case 1

Parameter	**Values**
Input	[4.000000, 8.000000, 12.000000]
Label (Expected Output)	[24.000000]
Predicted Output	[23.998917]
Error	[0.001083]
Status	Try Again

Test Case 2

Parameter	**Values**
Input	[4.000000, 8.000000, 12.000000]
Label (Expected Output)	[24.000000]
Predicted Output	[23.999160]
Error	[0.000840]

Parameter	Values
Status	Training Successful

Test Case 3

Parameter	Values
Input	[5.000000, 10.000000, 15.000000]
Label (Expected Output)	[30.000000]
Predicted Output	[29.999185]
Error	[0.000815]
Status	Training Successful

RESULT

- The neural network **successfully learned to compute the sum of input sequences** through supervised learning.
- **Predicted outputs closely matched** the actual target sums for both training and unseen test inputs.
- **Increasing the hidden layer neuron count** and **optimizing the learning rate** led to noticeable improvements in accuracy, faster convergence, and better overall stability during training.
- **Training error decreased consistently** across epochs, confirming effective weight updates and generalization capability.

CONCLUSION

The neural network effectively captured the summation pattern through supervised learning. Careful tuning of hyperparameters such as the learning rate, hidden layer size, and weight initialization significantly enhanced performance and reduced training error.