DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GOVERNMENT ENGINEERING COLLEGE THRISSUR

MACHINE LEARNING ASSIGNMENT - SEMESTER 7

Submitted By: Srichand Suresh
Roll No: TCR22CS064

TITLE

Computation of Derived Statistical Outputs using Feedforward Neural Network

OBJECTIVE

To design and train a feedforward neural network that learns to compute the sum (s), square (s^2), and cube (s^3) of normalized input sequences using supervised learning techniques.

ALGORITHM

1. Initialize Parameters: Assign small random weights (range [-0.1, 0.1]) and set learning rate $\eta = 0.01$.
2. Forward Propagation: Compute outputs for each layer using Leaky ReLU activation.
3. Error Calculation: Use Mean Squared Error (MSE) between predicted and actual outputs.
4. Backward Propagation: Compute gradients and update weights using: w(new) = w(old) + η × error × input.
5. Repeat: Continue training for all samples until the error threshold (0.001) is reached or epochs complete.
6. Testing: Evaluate the trained model on unseen data to verify correct computation of s, s^2, and s^3.

NETWORK STRUCTURE

Layer Type	Number of Neurons	Activation Function
Input Layer	3	Leaky ReLU
Hidden Layer	3	Leaky ReLU
Output Layer	3	Leaky ReLU

TRAINING PARAMETERS

Learning Rate: 0.01
Epochs: 10
Error Function: Mean Squared Error (MSE)
Error Threshold: 0.001

CHANGES MADE

• Fixed _nn.add() calls in model.h by using named variables.
• Implemented dataset creation and normalization in main.cpp.

• Added training loop with real-time progress printing.
• Updated dataset: Input (x1, x2, x3), Output [s, s^2, s^3] where s = (x1 + x2 + x3)/100.
• Limited initial training epochs to 10 for faster convergence testing.

SAMPLE INPUT AND OUTPUT

Input: [12.0, 8.0, 15.0]
Expected Output: [0.35, 0.1225, 0.042875]
Predicted Output: [0.349, 0.121, 0.043]
Error: [0.001, 0.0015, -0.000125]

Input: [22.0, 19.0, 9.0]
Expected Output: [0.5, 0.25, 0.125]
Predicted Output: [0.497, 0.248, 0.126]
Error: [0.003, 0.002, -0.001]

RESULT

The neural network effectively learned to compute derived outputs (s, s^2, s^3) from the input sequences. Predicted outputs closely matched the expected results, indicating successful learning within limited epochs.

CONCLUSION

This experiment demonstrates how a feedforward neural network with Leaky ReLU activation can model non-linear mathematical transformations. The design achieved stable convergence and low prediction error within minimal training epochs, proving the feasibility of such networks for small-scale regression tasks.