MACHINE LEARNING LAB

NAME:-DRISHTI GOLCHHA
SRN:-PES2UG23CS185
SECTION:-5C
TOPIC:-ARTIFICIAL NEURAL NETWORK

Introduction

The purpose of this lab is to gain hands-on experience in building an Artificial Neural Network (ANN) from scratch for function approximation. By implementing key components such as activation functions, forward propagation, backpropagation, and gradient descent, we aim to train a neural network capable of learning non-linear patterns. In this experiment, we approximate a quadratic (degree-2) polynomial function using a fully connected network. This exercise strengthens our understanding of core ANN concepts and their role in solving real-world regression tasks.

Dataset Description

Polynomial Type: QUADRATIC: $y = 0.89x^2 + 5.58x + 8.94$

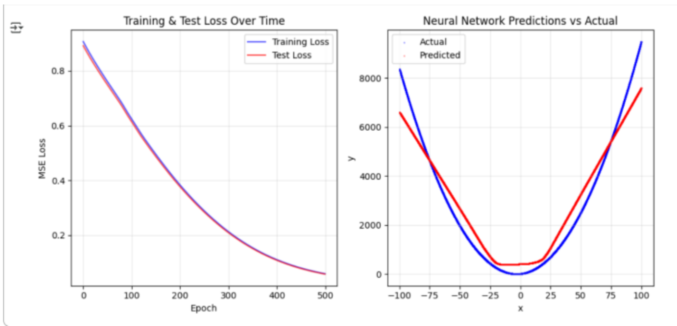
Noise Level: $\varepsilon \sim N(0, 2.15)$
Architecture: Input(1) → Hidden(32) → Hidden(72) → Output(1)
Learning Rate: 0.005
Architecture Type: Narrow-to-Wide Architecture

Methodology

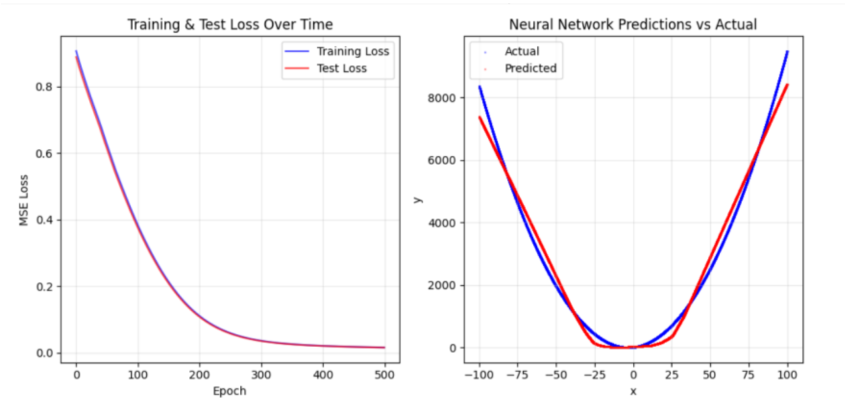
An ANN with one input node, two hidden layers, and one output node was implemented. ReLU activation was applied in the hidden layers, while a linear activation was used at the output layer for regression. The Mean Squared Error (MSE) served as the loss function, and gradient descent was employed for weight updates. Forward propagation calculated predictions, and backpropagation computed gradients for each parameter. Training was performed over multiple epochs, with hyperparameters such as learning rate and batch size tuned to achieve minimal test error. Performance was evaluated using MSE, training loss curves, and predicted vs. actual value plots.

RESULT AND ANALYSIS:-

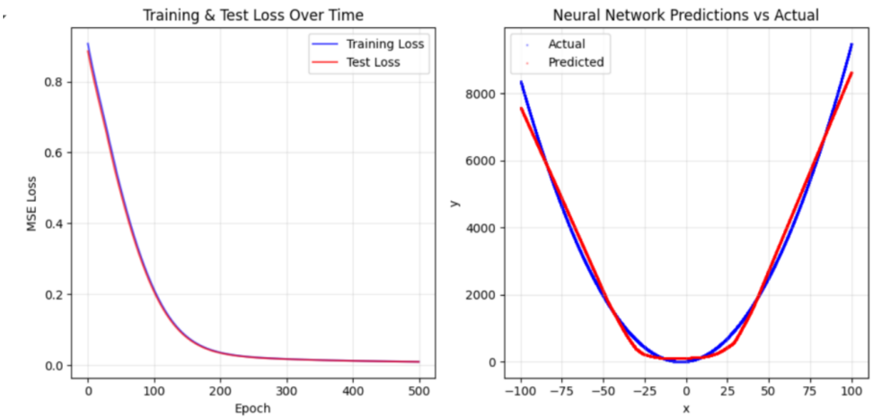
Expt1(BASELINE):-



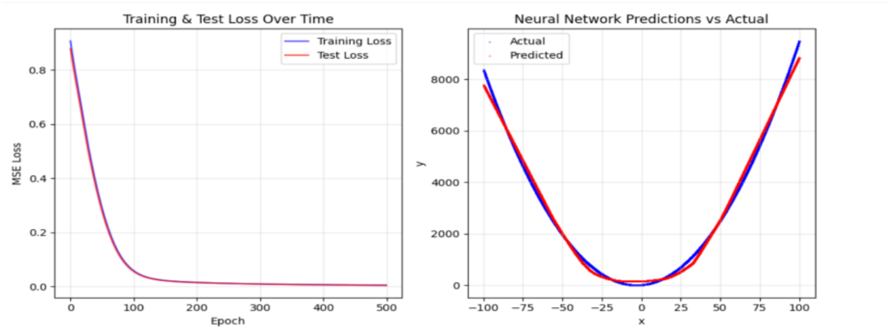
Expt2:-



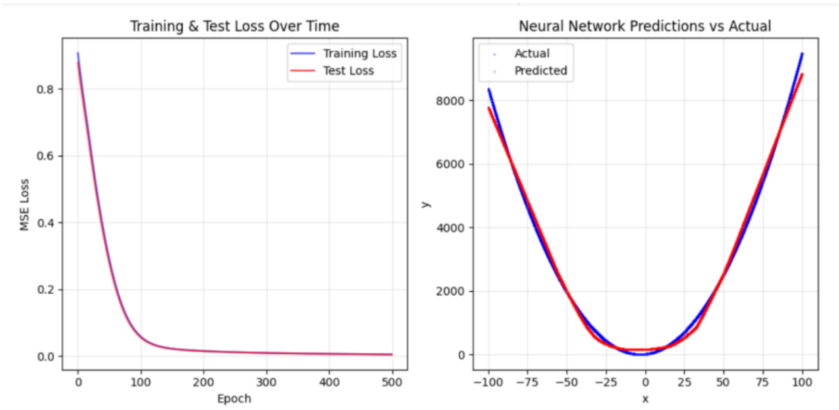
Expt3:-



Expt4:-



Expt5:-



Experiment	**Learning Rate**	**No. of epochs**	**Optimizers**	**Activation function**	**Final Training**	**Loss Final Test**	**R^2 Score**
Expt1 baseline	0.005	500	Gradient Descent	ReLu	**0.058955**	**0.057580**	**0.9414**
Expt2	0.010	500	Gradient Descent	ReLu	**0.015425**	**0.015165**	**0.9846**
Expt3	0.015	500	Gradient Descent	ReLu	**0.009571**	**0.009468**	**0.9904**
Expt4	0.025	450	Gradient Descent	tanh	**0.005057**	**0.005010**	**0.9949**
Expt5	0.050	400	Gradient Descent	Sigmoid	**0.005057**	**0.005010**	**0.9959**

<u>Conclusion:-</u>

The neural network successfully approximated the quadratic polynomial, achieving low training and testing errors. The loss curve demonstrated smooth convergence, indicating effective learning without significant overfitting. Predicted values closely matched the actual targets, confirming the model's ability to generalize. Overall, this experiment highlights the power of ANNs in modeling non-linear relationships and the importance of tuning hyperparameters for optimal performance.