MACHINE LEARNING
LAB-6

Project Title: Artificial Neural Networks

Name: Nehal G

SRN: PES2UG23CS380

Course: UE23CS352A: MACHINE LEARNING

Date: 19/09/2025

1.Introduction

Purpose of the Lab:

The purpose of this lab is to do a project on Neural Networks for Function Approximation. The objective of this assignment is to give you hands-on experience implementing a neural network from scratch, without relying on high-level frameworks like TensorFlow or PyTorch. We do this by analyzing polynomial regression for predictive modeling. By applying regression on the given dataset, we aim to evaluate the effect of polynomial features, noise, and regularization on the model's performance.

Tasks Performed:

• Generated a custom dataset

• Implemented the core components of a neural network: activation functions, loss functions, forward pass, backpropagation, and weight updates.

• Trained neural network to approximate the generated polynomial curve.

• Evaluated and visualized the performance of the model.

2. Dataset Description

• Dataset Type: Generated using SRN – Synthetic polynomial curve

• Samples: 100,000 (80,000 training, 20,000 testing)

• Features: 1 input feature (x), 1 output target (y)

• Noise: Gaussian noise with standard deviation assigned in dataset generation step

• Preprocessing: StandardScaler applied to both inputs and outputs

3. Methodology

• Architecture: Input (1), Hidden Layer 1 (ReLU), Hidden Layer 2 (ReLU), Output (1)

• Activation Function: ReLU for hidden layers, Linear for output

• Loss Function: Mean Squared Error (MSE)

• Optimizer: Gradient Descent

• Evaluation Metrics: Training/Test Loss, R^2 Score

4. Results and Analysis

Training Loss Curve

• Across all experiments, the training loss consistently decreased with epochs.
• With LR = 0.001 (Exp 1), the loss decreased slowly and plateaued at a high value (~0.67), showing very slow convergence.
• With LR = 0.01 (Exps 2, 3, 5), the loss reduced rapidly to very small values (~0.01–0.04), showing that the model learned effectively.
• With LR = 0.005 (Exp 4), convergence was good but slower than LR = 0.01, stabilizing around ~0.056.
• The best loss curve was observed in Exp 5 (LR=0.01, 700 epochs), where the model steadily reached ~0.011.

Final Test MSE

• Exp 1 (LR=0.001, 500 epochs): Test Loss = 0.6708 . Very high error, poor performance.
• Exp 2 (LR=0.01, 500 epochs): Test Loss = 0.0184 .Huge improvement, excellent fit.
• Exp 3 (LR=0.01, 300 epochs): Test Loss = 0.0381 . Slightly worse than Exp 2, but still strong.
• Exp 4 (LR=0.005, 500 epochs): Test Loss = 0.0548 .Good but not as strong as Exp 2.
• Exp 5 (LR=0.01, 700 epochs): Test Loss = 0.0110 . Best performance (lowest error).

Plot of Predicted vs Actual Values

• Exp 1 (LR=0.001): Predicted curve diverges from actual, showing underfitting.
• Exp 2 & Exp 3 (LR=0.01): Predicted values align very closely with actual curve, almost overlapping.
• Exp 4 (LR=0.005): Predictions are close but with slightly more deviation than Exp 2.
• Exp 5 (LR=0.01, 700 epochs): Predictions almost perfectly overlap with the true curve.

Discussion on Performance

• Underfitting was observed in Exp 1, where the model could not capture the data complexity due to too small a learning rate. Loss remained high.
• Good fits were achieved in Exps 2, 3, and 4, where learning rate and epochs were

balanced. R^2 scores were above 0.94, indicating strong correlation.
• Best fit occurred in Exp 5, with R^2 = 0.9890, where both training and test loss were minimized. The model generalized well without overfitting.
• No major overfitting signs: In all experiments, training and test losses were very close, meaning the model learned general patterns rather than memorizing.

Results Table

Exp	Learning Rate	Epochs	Train Loss	Test Loss	R^2 Score
1	0.001	500	0.6711	0.6708	0.3308
2	0.01	500	0.0188	0.0184	0.9816
3	0.01	300	0.0391	0.0381	0.9620
4	0.005	500	0.0562	0.0548	0.9454
5	0.01	700	0.0112	0.0110	0.9890

5. Conclusion
• Best Model: Learning Rate = 0.01, Epochs = 700 → achieved lowest Test Loss (0.0110) and highest R^2 (0.9890).

• Underfitting observed when learning rate was too small (LR=0.001).

• Overfitting was not significant; test loss remained close to training loss across runs.

• Increasing epochs beyond 500 improved performance when learning rate was appropriately set.

• Key Takeaway: Proper tuning of learning rate and epochs is crucial. Too low LR causes underfitting, while balanced LR (0.01) with higher epochs (700) gives the best approximation.