Ali Alsharif
CSC 671

Implementing and Evaluating a MLP for Regression

The objective of this assignment is to implement a Multi-Layer Perceptron (MLP) for a regression task, using the dataset provided in mlp_regression_data.csv. This involves creating an MLP model in PyTorch, normalizing the data, defining appropriate loss functions, training the model, and evaluating its performance through plotting the regression curve and analyzing training loss across epochs.

Data Preparation

We begin by loading the dataset from mlp_regression_data.csv, which consists of 1000 data points. Each data point has an independent variable 'x' and a dependent variable 'y'. The data is then normalized using z-score standardization to ensure that our MLP model receives inputs that are on a similar scale.

MLP Model Design

The MLP model consists of three linear layers: two hidden layers and one output layer. The model architecture is as follows:

- Input Layer: Receives the single feature 'x'.
- First Hidden Layer: Consists of 50 neurons.
- Second Hidden Layer: Consists of 25 neurons.
- Output Layer: Outputs a single value, representing the predicted 'y'.

Loss Function and Optimization

The Mean Squared Error (MSE) loss function is used for measuring the difference between the predicted and actual values in a regression. The model is optimized using Stochastic Gradient Descent with a learning rate of 0.01.

Training Process

The model is trained over 100 epochs, with the dataset loaded into a DataLoader to batch process the data points. The training process involves forward propagation, loss calculation, backpropagation, and optimization steps for each batch of data.

Evaluation and Analysis

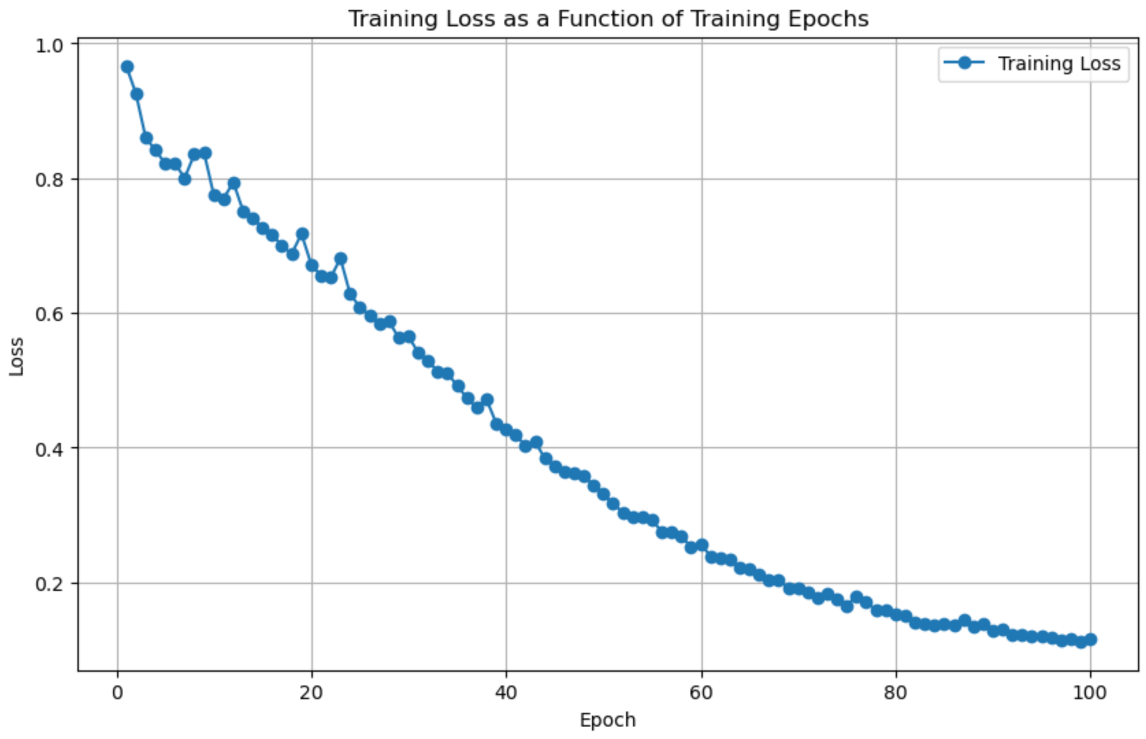
The model's performance is evaluated by plotting the regression curve against the actual data points. This visual representation helps in understanding how well the MLP model has learned the underlying pattern of the data. Additionally, the training loss is plotted across epochs to analyze the model's learning progress over time.

Conclusion

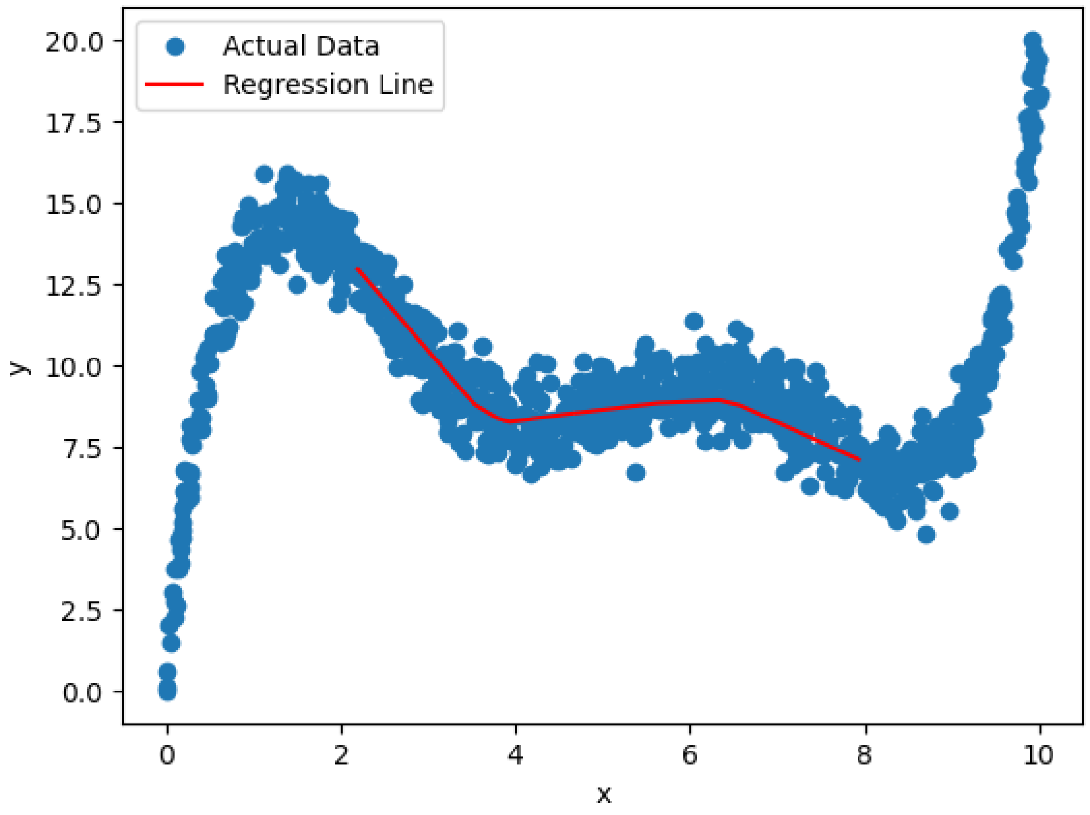
The implementation of the MLP for regression demonstrates the capability of neural networks to model complex, non-linear relationships between variables. The analysis of training loss across epochs reveals the model's learning efficiency and convergence behavior.

<u>Plots in next page</u>

Training loss:



Regression curve:


Actual Data
Regression Line
y
x