MLPR Lab-12

This assignment demonstrates the training and evaluation of a neural network for fertility classification using a single hidden layer. It covers key concepts such as feedforward pass, backpropagation, error calculation, and tuning hyperparameters.

**Instructions:**

* Submit the graph along with the complete code.
* Using **Google Colab is recommended** for this lab as there might some issues occurs due to the tensorflow installation and their dependencies.
* **Approx. program execution time – 6 to 10 minutes**

**Step1:** Import libraries

* Numpy
* Pyplot
* From sklearn.metrics import mean\_squared\_error
* Tensorflow

**Step2**: Load “Fertility\_Diagnosis.txt” file. The data has 10 attributes, the first 9 of them being the features and the last one being the target variable.

* Use *np.genfromtxt()*
* Create training and testing datasets.

**Step3**: Define input layer size and output layer size.

* Create an empty list to store training error and testing error separately.
* Keep error tolerance 0.05.

**Step4**: Create a loop for having different no. of neurons (from 1 to 9) in the hidden layer. Under this loop, do the following-

* Initialize random synapse weights.
* Define feedforward network for training data.
  + Layer 0 – Input data
  + Layer 1- Sigmoid activation Function
  + Layer 2- Sigmoid activation Function
* Check for accuracy.
  + Calculate mean square error.
  + Print MSE
* Keep learning rate (alpha) = 0.001
* Print – training error with alpha value.

**Step5**: For range 1 to 1000001 perform backpropagation.

* Define feedforward network same as in step 4.
* Define cost function (Difference between training data outputs and layer 2)
* Calculate the derivative of the sigmoid activation with respects to the weighted sum of inputs at layer 2. This derivative can be used in backpropagation algorithm for updating the weights.
* Update the weight using calculated derivative.
* Now calculate the derivative of the sigmoid activation with respects to the weighted sum of inputs at layer 1.

**Step6**: Check the convergence and print the debug data.

* If error value is less than error tolerance value, break the loop and print the stopping at value.
* Print the debug data during training.

**Step7**: Evaluate the training results.

* Perform a feedforward pass through the trained network on the training data.
* Calculate and print the mean squared error for the training dataset.
* Append the training error to the list.

**Step8**: Evaluate testing results.

* Perform a feedforward pass through the trained network on the testing data.
* Calculate and print the mean squared error for the testing dataset.
* Append the testing error to the list.

**Step9**: Plot the results.

* Plot the training and testing errors for different numbers of neurons in the hidden layer using Matplotlib.
* Reference output given below

A graph of a graph with lines and numbers

Description automatically generated with medium confidence

QUESTIONS

1. What happens if all the weights are initialized to zero (zero initialization)?
2. What is the difference between epoch, batch, and iteration in neural networks?
3. What is the key difference between stochastic gradient decent and batch gradient decent?
4. What problems does a completely random initialization of a neural network lead to?
5. Name any ten hyperparameters in a multi-layer neural network.