Lab 4 - Neural Networks

In this lab, We implement different configurations of neural network to approximate the function $y = 0.2x^4 + 2x^3 + 0.1x^2 + 10$ where x is real between -1 and 1 inclusive ($-1 \leq x \leq 1$).
Note: you can use any standard available libraries to finish the lab.

Implementation steps

- Create $30,000$ samples which include x and associated y values.

  Q1. Write a function to create x and y values that follows the given function above. Provide initial plot of the function.

- Write a function that accepts an argument to indicate shuffling. Based on the argument, function has to return the shuffled data or data without shuffling.

  Q2. Submit the above function in your code.

  hint:

```
get_dataset(argument):
    generate x data
    generate y data

    if argument = 'shuffle'
        shuffle the data

return data
```

- Write a function to split the data into train, validation, and test sets. The function should be able to accept the data and ratios of train, validation, and test data.

  Q3. Submit the above function in your code.

- Write a function to scale the data between 0 and 1.

  Q4. Submit the above function in your code.

- Write a function that accepts actual target and predicted values to calculate mean absolute error (MAE), mean square error(MSE), root mean square error (RMSE), and r2 score

  Q5. Submit the above function in your code.

- In this lab, we will implement neural network of two different structures and using different hyper-parameters. We use fully connected layer (FCL), also termed as Dense in Tensorflow. "Relu" and "tanh" activation functions. "mse" as loss function and "adam" as optimizer.

  Structure 1:

```
FCL(12 units) // first hidden layer
FCL(8 units) // second hidden layer
FCL(4 units) // last hidden layer
```

  Structure 2:

```
FCL(24 units) // one hidden layer
```

- Implement the following cases. Provide graphs and metrics as mentioned. In all cases use the followings.
  - **Data split ratios:** 30% training set, 20% validation set, and 50% test set.
  - **Number of epochs**: 20
  - **Bath size**: 12
  - **Loss function**: MSE
  - **Optimizer**: Adam

  Provide two separate graphs having actual test data (x_test vs. y_test) and predicted data (x_test vs. y_predicted). MAE, MSE, RMSE, and R2 score for all cases.

  Case 1:

  - **Data:** Use shuffled and unscaled data.
  - **NN structure:** Structure 1.
  - **Activation function:** Relu.

  Case 2:

  - **Data:** Use shuffled and unscaled data.
  - **NN structure:** Structure 2.
  - **Activation function:** Relu.

  Case 3:

  - **Data:** Use shuffled and unscaled data.
  - **NN structure:** Structure 1.
  - **Activation function:** tanh.

  Case 4:

  - **Data:** Use shuffled and scaled data. First, shuffle the data then split the data and finally scale the data (both x and y).
  - **NN structure:** Structure 1.
  - **Activation function:** Relu.

  Case 5:

  - **Data:** Use shuffled and scaled data. First, shuffle the data then split the data and finally scale the data (both x and y).
  - **NN structure:** Structure 1.
  - **Activation function:** tanh.

  Repeat all the above five cases for data that not shuffled.

- **Q6** Solve the XOR problem by using a neural network. Design your neural network by assuming all required parameters, such as the number of layers, activation functions, loss function, etc. Present your work to classify the given truth table with sufficient details.

Truth Table

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

NOTE: Available APIs should not be used to answer Q6. All required functions should be implemented from scratch

Lab submission

- Submit only .ipynb file.
- Students need to be present in the lab for an in-person demonstration. Your TA might ask a few questions.
- Submissions are accepted by uploading to D2L; email submissions will be ignored.
- Late submissions will receive a 10% penalty per day up to 3 days. After 72 hours from the deadline, 0 will be assigned.
- If the code will not run, 0 will be assigned. Mention the platform specifics in the code if any. Any syntax errors that stall the execution will be assigned with 0. All required libraries should be installed in the code itself.
- Test your code to see if it executes appropriately in google colab (for .ipynb) before submitting.