ADTA 5550.401: Deep Learning with Big Data

Biniam Abebe

**Assignment 3**

1. PART I: One-Hot Encoding

### Question 1.1:

Using critical thinking and based on the lectures, is it necessary to perform any coding (integer coding or one-hot coding) on these class values for better project performance?

Answer: Yes, encoding is necessary for the class values to improve the deep learning project's performance.

### Question 1.2:

If the answer to Question 1.1 is “YES,” what type(s) of encoding needs to be done to process the class values before using the dataset for the deep learning project?

Answer: One-hot encoding is required for the class values.

### Question 1.3:

Based on the answer to Question 1.2, explain what needs to be done for each encoding type.

Answer:

* Identify the class attribute and its possible values.
* Use a library like Pandas or Scikit-Learn in Python to perform one-hot encoding.
* Ensure the encoded values are in a format compatible with the neural network.

### Question 1.4:

Based on the answer to Question 1.2, perform the necessary encoding tasks to transform the class values before using the dataset for the deep learning project.

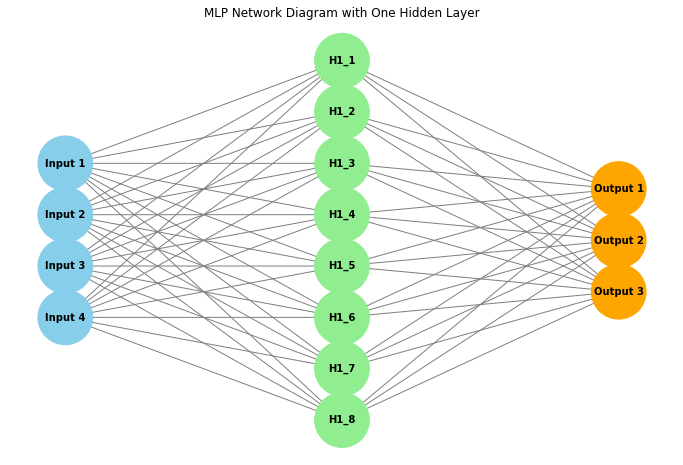
Answer:

* Load the dataset (Iris.csv) into a Pandas DataFrame.
* Apply one-hot encoding to the class attribute.
* Save and display the encoded dataset.
* Functions:
  + encode\_class\_values: Encodes the class values of the given data using LabelEncoder.
  + convert\_to\_one\_hot: Converts the encoded class values to one-hot encoding using np\_utils.to\_categorical.
* Variables:
  + encoder\_train: LabelEncoder object for training data.
  + encoder\_test: LabelEncoder object for testing data.
  + encoded\_Y\_train: Encoded class values of the training data.
  + encoded\_Y\_test: Encoded class values of the testing data.
  + onehot\_Y\_train: One-hot encoded class values of the training data.
  + onehot\_Y\_test: One-hot encoded class values of the testing data.

1. PART II: MLPs (Fully Connected Neural Networks) with Keras

The Multi-Layer Perceptron (MLP) design consists of three main layers: an input layer, one hidden layer, and an output layer. The input layer has four neurons, each representing one of the features in the Iris dataset (sepal length, sepal width, petal length, and petal width). The hidden layer has eight neurons and uses the ReLU activation function, which helps the model learn complex patterns in the data by applying non-linear transformations. Finally, the output layer has three neurons, corresponding to the three classes of the Iris dataset (Iris Setosa, Iris Versicolour, Iris Virginica), and uses the sigmoid activation function to produce probabilities for each class.

The design choice of having one hidden layer with eight neurons balances complexity and efficiency, ensuring the model has enough capacity to learn without overfitting. The ReLU activation function is chosen for its effectiveness in training deep networks, while the sigmoid function in the output layer is suitable for generating class probabilities. The model is trained using the Adam optimizer and categorical cross-entropy loss function, which are well-suited for multi-class classification tasks.



This simple structure allows the MLP to effectively learn from the Iris dataset and make accurate predictions.

1. PART III: Redesign the MLP (30 Points)

**A network of circles and lines

Description automatically generated with medium confidence**

### Explanation:

1. **Enhanced Model Definition:**
   * The enhanced\_model function defines a neural network with two hidden layers.
   * The first hidden layer has 8 neurons with ReLU activation.
   * The second hidden layer is added with 8 neurons and ReLU activation.
   * The output layer has 3 neurons with sigmoid activation for multi-class classification.
2. **Training and Evaluation:**
   * The model is compiled using the Adam optimizer and categorical cross-entropy loss.
   * The model is trained on the training data.
   * The model's accuracy is evaluated on the test data and printed.

This updated model includes an additional hidden layer to potentially improve its ability to capture complex patterns in the data.