1. **Dataset Generation**:
    - The **make_classification** function from **sklearn.datasets** is used to generate a synthetic dataset with 100 samples, 8 input features, and 4 output classes.
    - This dataset generation ensures that we have both input features (**X**) and corresponding output labels (**y**) for training the neural network.
2. **One-Hot Encoding**:
    - Since this is a multi-class classification problem, the output labels (**y**) are encoded using one-hot encoding.
    - The **OneHotEncoder** from **sklearn.preprocessing** is used for this purpose.
3. **Neural Network Architecture**:
    - The neural network architecture consists of an input layer with 8 neurons (to match the number of features), one hidden layer with 10 neurons, and an output layer with 4 neurons (to match the number of classes).
    - The weights (**W_HI_1** and **W_O_1**) between layers are initialized randomly.
4. **Feedforward Propagation**:
    - In the **feedForward** method, the input data (**X**) is propagated forward through the network.
    - The hidden layer output (**hidden_output_1**) is computed using the sigmoid activation function.
    - The output layer predictions (**pred**) are computed similarly using the sigmoid activation function.
5. **Backpropagation**:
    - The **backPropagation** method implements the backpropagation algorithm to update the weights based on the error between the predictions and true labels.
    - It calculates the output error, output delta, hidden error, and hidden delta.
    - Finally, it updates the weights (**W_HI_1** and **W_O_1**) using gradient descent.

6. **Training**:
    - The neural network is trained for 5000 iterations using the training data (**X_train** and **y_train_encoded**).
    - The training error (mean squared error) is plotted to monitor the training progress.
7. **Testing and Evaluation**:
    - The trained model is tested on the test data (**X_test**).
    - Model output (probabilities for each class) is printed, followed by the one-hot encoded predictions and the predicted class labels.
    - The accuracy of the model on the test data is calculated.
    - A confusion matrix is computed and printed to evaluate the model's performance further.

Challenges Faced:

- One challenge might be understanding the concepts of multi-class classification and implementing one-hot encoding for the output labels. However, the code handles this smoothly by using the **OneHotEncoder** provided by scikit-learn.
- Another challenge could be debugging the neural network implementation, especially the backpropagation algorithm. However, the code includes comments and clear variable names to aid in understanding and debugging.

Performance and Observations:

- The neural network achieves decent accuracy on the test data, which is a good sign.
- The training error decreases over iterations, indicating that the model is learning from the training data.
- The confusion matrix provides insights into the model's performance for each class, helping identify any specific classes where the model struggles.
- Further optimizations and tuning, such as adjusting the learning rate or adding more layers/neurons, could potentially improve the model's performance.