Batch: B1

Roll Number: 16010421073 **Experiment Number: 3**

Name: Keyur Patel

Title of the Experiment: Deep Neural Network

Program:

```
import numpy as np
import tensorflow as tf
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from tensorflow.keras import models, layers, optimizers


# Step 1: Download the required dataset
iris = load_iris()
X = iris.data
y = iris.target


# Step 2: Define the Input Shape
input_shape = X.shape[1]


# Step 3: Create a Model with no of hidden layer and output layer
def create_model(hidden_units=16, activation='relu'):
    model = models.Sequential()
    model.add(layers.Dense(hidden_units, activation=activation, input_shape=(input_shape,)))
    model.add(layers.Dense(3, activation='softmax'))  # Output layer with 3 units for 3 classes
    return model
```

```
# Step 4: Decide the values of hyperparameters
hidden_units = 16
activation = 'relu'
learning_rate = 0.001
batch_size = 32
epochs = 50

# Step 5: Try to use different activation functions
activation_functions = ['relu', 'tanh', 'sigmoid']

for activation in activation_functions:
    # Step 6: Analyze the effect of various values of hyperparameters
    model = create_model(hidden_units=hidden_units,
activation=activation)
    model.compile(optimizer=optimizers.Adam(learning_rate=learning_rate
),
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])

    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

    model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size,
verbose=0)

    loss, accuracy = model.evaluate(X_test, y_test, verbose=0)
    print(f"Activation: {activation}, Test Accuracy: {accuracy}")

model.summary()
```

Output:

```
Activation: relu, Test Accuracy: 0.8666666746139526
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_test_functio
Activation: tanh, Test Accuracy: 0.9666666388511658
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_test_functio
Activation: sigmoid, Test Accuracy: 0.800000011920929
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_10 (Dense)            (None, 16)                80

 dense_11 (Dense)            (None, 3)                 51

=================================================================
Total params: 131 (524.00 Byte)
Trainable params: 131 (524.00 Byte)
Non-trainable params: 0 (0.00 Byte)
```


Post Lab Question- Answers (If Any):

1. **Deep learning works well despite of __problem(s).**
   a. Sharp Minima
   b. Numerical instability (vanishing/exploding gradient)
   c. High capacity (susceptible to overfitting)
   d. All of the above

2. **The number of neurons in the output layer should match the number of classes (where no of classes are greater than 2) in a supervised learning task. True or False?**
   a. True
   b. False

3. List down activation function functions most widely used at hidden layer and output layer.

Hidden Layer:

ReLU (Rectified Linear Unit): $f(x) = \max(0,x)$

Sigmoid: $f(x) = 1 / 1+e^{-x}$

Tanh (Hyperbolic Tangent): $f(x) = e^{x} - e^{-x} / e^{x} + e^{-x}$

Leaky ReLU: $f(x) = \{x, \quad \text{if } x>0$

$0.01x, \text{otherwise}\}$

ELU (Exponential Linear Unit): $= f(x) = \{x, \quad \text{if } x>0.$

$\alpha \cdot (ex-1), \text{otherwise}$

Output Layer:

Sigmoid: Typically used for binary classification tasks to output probabilities between 0 and 1.

Softmax: Generally used for multi-class classification tasks to output probability distributions over multiple classes.

Linear: Used for regression tasks when the output is a continuous value without any activation function applied.

Conclusion: In this experiment, we learnt about deep neural networks, implemented the same and printed the model summary.
