**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)

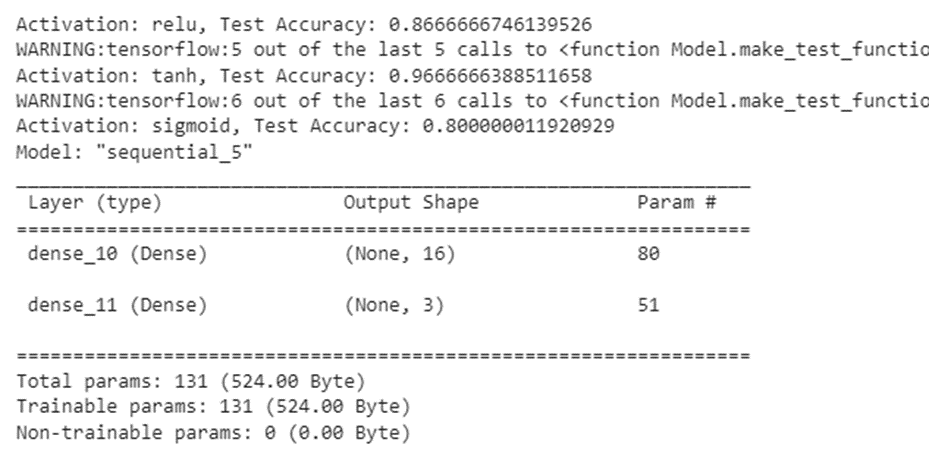
    model.fit(X\_train, y\_train, epochs=epochs, batch\_size=batch\_size, verbose=0)

    loss, accuracy = model.evaluate(X\_test, y\_test, verbose=0)

    print(f"Activation: {activation}, Test Accuracy: {accuracy}")

model.summary()

**Output:**

****

**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) = ex - e−x / ex + e−x​

**Leaky ReLU:** f(x) = {x, if x>0

0.01x, otherwise}

**ELU (Exponential Linear Unit):** = f(x) = {x, if x>0.

α⋅(ex−1),​ 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.

**CO2:** Comprehend the Deep Network concepts.

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