NEURAL NETWORKS ASSIGNMENT2  
  
 NAME: Saharsha Muddagauni  
  
 Std ID: 700742679

Git hub link : https://github.com/sxm26790/Assignment2-

Problem 1:

1. Use the use case in the class: a. Add more Dense layers to the existing code and check how the accuracy changes.

from google.colab import drive  
drive.mount('/content/gdrive')

Mounted at /content/gdrive

path\_to\_csv = '/content/gdrive/My Drive/diabetes.csv'

import keras  
import pandas  
from keras.models import Sequential  
from keras.datasets import mnist  
from keras.layers.core import Dense, Activation  
  
# load dataset  
from sklearn.model\_selection import train\_test\_split  
import pandas as pd  
import numpy as np  
  
dataset = pd.read\_csv(path\_to\_csv, header=None).values  
  
X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(dataset[:,0:8], dataset[:,8],  
 test\_size=0.25, random\_state=87)  
np.random.seed(155)  
my\_first\_nn = Sequential() # create model  
my\_first\_nn.add(Dense(20, input\_dim=8, activation='relu')) # hidden layer  
my\_first\_nn.add(Dense(4, activation='relu')) # hidden layer  
my\_first\_nn.add(Dense(1, activation='sigmoid')) # output layer  
my\_first\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])  
my\_first\_nn\_fitted = my\_first\_nn.fit(X\_train, Y\_train, epochs=100,  
 initial\_epoch=0)  
print(my\_first\_nn.summary())  
print(my\_first\_nn.evaluate(X\_test, Y\_test))

Epoch 1/100  
18/18 [==============================] - 1s 4ms/step - loss: 4.8242 - acc: 0.3403  
Epoch 2/100  
18/18 [==============================] - 0s 4ms/step - loss: 2.2425 - acc: 0.4115  
Epoch 3/100  
18/18 [==============================] - 0s 5ms/step - loss: 1.2439 - acc: 0.5330  
Epoch 4/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.7845 - acc: 0.6076  
Epoch 5/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6986 - acc: 0.6528  
Epoch 6/100  
18/18 [==============================] - 0s 21ms/step - loss: 0.6858 - acc: 0.6649  
Epoch 7/100  
18/18 [==============================] - 0s 17ms/step - loss: 0.6829 - acc: 0.6667  
Epoch 8/100  
18/18 [==============================] - 0s 14ms/step - loss: 0.6801 - acc: 0.6667  
Epoch 9/100  
18/18 [==============================] - 0s 10ms/step - loss: 0.6780 - acc: 0.6649  
Epoch 10/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6764 - acc: 0.6667  
Epoch 11/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6746 - acc: 0.6615  
Epoch 12/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6729 - acc: 0.6615  
Epoch 13/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6713 - acc: 0.6580  
Epoch 14/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6690 - acc: 0.6667  
Epoch 15/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6673 - acc: 0.6667  
Epoch 16/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6660 - acc: 0.6667  
Epoch 17/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6649 - acc: 0.6667  
Epoch 18/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6636 - acc: 0.6649  
Epoch 19/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6626 - acc: 0.6667  
Epoch 20/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6615 - acc: 0.6649  
Epoch 21/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6603 - acc: 0.6649  
Epoch 22/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6593 - acc: 0.6667  
Epoch 23/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6585 - acc: 0.6649  
Epoch 24/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6571 - acc: 0.6667  
Epoch 25/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6560 - acc: 0.6684  
Epoch 26/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6551 - acc: 0.6667  
Epoch 27/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6540 - acc: 0.6684  
Epoch 28/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6531 - acc: 0.6667  
Epoch 29/100  
18/18 [==============================] - 0s 5ms/step - loss: 0.6523 - acc: 0.6701  
Epoch 30/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6512 - acc: 0.6701  
Epoch 31/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6503 - acc: 0.6701  
Epoch 32/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6495 - acc: 0.6701  
Epoch 33/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6487 - acc: 0.6701  
Epoch 34/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6481 - acc: 0.6684  
Epoch 35/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6471 - acc: 0.6719  
Epoch 36/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6464 - acc: 0.6719  
Epoch 37/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6456 - acc: 0.6719  
Epoch 38/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6449 - acc: 0.6736  
Epoch 39/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6442 - acc: 0.6736  
Epoch 40/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6436 - acc: 0.6736  
Epoch 41/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6429 - acc: 0.6736  
Epoch 42/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6422 - acc: 0.6753  
Epoch 43/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6418 - acc: 0.6753  
Epoch 44/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6411 - acc: 0.6753  
Epoch 45/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6406 - acc: 0.6753  
Epoch 46/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6403 - acc: 0.6701  
Epoch 47/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6386 - acc: 0.6719  
Epoch 48/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6378 - acc: 0.6753  
Epoch 49/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6371 - acc: 0.6736  
Epoch 50/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6354 - acc: 0.6788  
Epoch 51/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6345 - acc: 0.6840  
Epoch 52/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6345 - acc: 0.6788  
Epoch 53/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6328 - acc: 0.6823  
Epoch 54/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6321 - acc: 0.6840  
Epoch 55/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6316 - acc: 0.6806  
Epoch 56/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6320 - acc: 0.6806  
Epoch 57/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6312 - acc: 0.6823  
Epoch 58/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6303 - acc: 0.6823  
Epoch 59/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6295 - acc: 0.6823  
Epoch 60/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6289 - acc: 0.6823  
Epoch 61/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6284 - acc: 0.6823  
Epoch 62/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6282 - acc: 0.6858  
Epoch 63/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6279 - acc: 0.6806  
Epoch 64/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6279 - acc: 0.6788  
Epoch 65/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6255 - acc: 0.6875  
Epoch 66/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6258 - acc: 0.6806  
Epoch 67/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6244 - acc: 0.6840  
Epoch 68/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6256 - acc: 0.6788  
Epoch 69/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6228 - acc: 0.6858  
Epoch 70/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6217 - acc: 0.6840  
Epoch 71/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6215 - acc: 0.6858  
Epoch 72/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6215 - acc: 0.6892  
Epoch 73/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6216 - acc: 0.6858  
Epoch 74/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6196 - acc: 0.6892  
Epoch 75/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6196 - acc: 0.6858  
Epoch 76/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6199 - acc: 0.6840  
Epoch 77/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6190 - acc: 0.6840  
Epoch 78/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6205 - acc: 0.6875  
Epoch 79/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6192 - acc: 0.6892  
Epoch 80/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6173 - acc: 0.6858  
Epoch 81/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6170 - acc: 0.6910  
Epoch 82/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6168 - acc: 0.6910  
Epoch 83/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6165 - acc: 0.6944  
Epoch 84/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6160 - acc: 0.6910  
Epoch 85/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6147 - acc: 0.6962  
Epoch 86/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6147 - acc: 0.6910  
Epoch 87/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6152 - acc: 0.6910  
Epoch 88/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6143 - acc: 0.6962  
Epoch 89/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6135 - acc: 0.6910  
Epoch 90/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6148 - acc: 0.6892  
Epoch 91/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6125 - acc: 0.6962  
Epoch 92/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6134 - acc: 0.6979  
Epoch 93/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6143 - acc: 0.6910  
Epoch 94/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6127 - acc: 0.6927  
Epoch 95/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6128 - acc: 0.6962  
Epoch 96/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6126 - acc: 0.6979  
Epoch 97/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6111 - acc: 0.6944  
Epoch 98/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6116 - acc: 0.6997  
Epoch 99/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6110 - acc: 0.6979  
Epoch 100/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6113 - acc: 0.6962  
Model: "sequential\_3"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 Layer (type) Output Shape Param #   
=================================================================  
 dense\_8 (Dense) (None, 20) 180   
   
 dense\_9 (Dense) (None, 4) 84   
   
 dense\_10 (Dense) (None, 1) 5   
   
=================================================================  
Total params: 269  
Trainable params: 269  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
None  
6/6 [==============================] - 0s 3ms/step - loss: 0.6150 - acc: 0.6719  
[0.6149560809135437, 0.671875]

Question 2(problem 1):

Change the data source to Breast Cancer dataset \* available in the source code folder and make required ***changes***. Report accuracy of the model.

from google.colab import drive  
drive.mount('/content/gdrive')

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force\_remount=True).

path\_to\_csv = '/content/gdrive/My Drive/breastcancer.csv'

import keras  
import pandas  
from keras.models import Sequential  
from keras.datasets import mnist  
from keras.layers.core import Dense, Activation  
from sklearn.datasets import load\_breast\_cancer  
from sklearn.model\_selection import train\_test\_split  
import pandas as pd  
import numpy as np  
  
# Load dataset  
  
cancer\_dataset = load\_breast\_cancer()  
X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(dataset[:,0:8], dataset[:,8],  
 test\_size=0.25, random\_state=87)  
  
np.random.seed(155)  
my\_first\_nn = Sequential() # create model  
my\_first\_nn.add(Dense(20, input\_dim=8, activation='relu')) # hidden layer  
my\_first\_nn.add(Dense(1, activation='sigmoid')) # output layer  
my\_first\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])  
my\_first\_nn\_fitted = my\_first\_nn.fit(X\_train, Y\_train, epochs=100,  
 initial\_epoch=0)  
print(my\_first\_nn.summary())  
print(my\_first\_nn.evaluate(X\_test, Y\_test))

Epoch 1/100  
18/18 [==============================] - 1s 2ms/step - loss: 41.4774 - acc: 0.6615  
Epoch 2/100  
18/18 [==============================] - 0s 2ms/step - loss: 30.9226 - acc: 0.6615  
Epoch 3/100  
18/18 [==============================] - 0s 2ms/step - loss: 20.9840 - acc: 0.6615  
Epoch 4/100  
18/18 [==============================] - 0s 2ms/step - loss: 11.4362 - acc: 0.6562  
Epoch 5/100  
18/18 [==============================] - 0s 2ms/step - loss: 3.8298 - acc: 0.6562  
Epoch 6/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.5091 - acc: 0.6111  
Epoch 7/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.1508 - acc: 0.6406  
Epoch 8/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.0482 - acc: 0.6441  
Epoch 9/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.9634 - acc: 0.6545  
Epoch 10/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.8632 - acc: 0.6424  
Epoch 11/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.7702 - acc: 0.6528  
Epoch 12/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.6834 - acc: 0.6684  
Epoch 13/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.6130 - acc: 0.6632  
Epoch 14/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.5201 - acc: 0.6649  
Epoch 15/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.4559 - acc: 0.6632  
Epoch 16/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.3763 - acc: 0.6597  
Epoch 17/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.3169 - acc: 0.6753  
Epoch 18/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.2442 - acc: 0.6753  
Epoch 19/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.1798 - acc: 0.6736  
Epoch 20/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.1353 - acc: 0.6753  
Epoch 21/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.0747 - acc: 0.6788  
Epoch 22/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.0069 - acc: 0.6736  
Epoch 23/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.9553 - acc: 0.6788  
Epoch 24/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9134 - acc: 0.6649  
Epoch 25/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9014 - acc: 0.6684  
Epoch 26/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.8562 - acc: 0.6562  
Epoch 27/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.8200 - acc: 0.6788  
Epoch 28/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7870 - acc: 0.6597  
Epoch 29/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7682 - acc: 0.6562  
Epoch 30/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7374 - acc: 0.6701  
Epoch 31/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7362 - acc: 0.6528  
Epoch 32/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7151 - acc: 0.6771  
Epoch 33/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7133 - acc: 0.6510  
Epoch 34/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6979 - acc: 0.6510  
Epoch 35/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7079 - acc: 0.6510  
Epoch 36/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6944 - acc: 0.6892  
Epoch 37/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7036 - acc: 0.6545  
Epoch 38/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7029 - acc: 0.6545  
Epoch 39/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6868 - acc: 0.6615  
Epoch 40/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6795 - acc: 0.6649  
Epoch 41/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6717 - acc: 0.6840  
Epoch 42/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6713 - acc: 0.6806  
Epoch 43/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7170 - acc: 0.6372  
Epoch 44/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6922 - acc: 0.6597  
Epoch 45/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6743 - acc: 0.6788  
Epoch 46/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6799 - acc: 0.6545  
Epoch 47/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6600 - acc: 0.6753  
Epoch 48/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6492 - acc: 0.6962  
Epoch 49/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6622 - acc: 0.6753  
Epoch 50/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6648 - acc: 0.6701  
Epoch 51/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6665 - acc: 0.6667  
Epoch 52/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6650 - acc: 0.6840  
Epoch 53/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6400 - acc: 0.7014  
Epoch 54/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6578 - acc: 0.6892  
Epoch 55/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6404 - acc: 0.6962  
Epoch 56/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6761 - acc: 0.6719  
Epoch 57/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6729 - acc: 0.6840  
Epoch 58/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6419 - acc: 0.6944  
Epoch 59/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6424 - acc: 0.6892  
Epoch 60/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6786 - acc: 0.6615  
Epoch 61/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6667 - acc: 0.6892  
Epoch 62/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6314 - acc: 0.6944  
Epoch 63/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6246 - acc: 0.6962  
Epoch 64/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6261 - acc: 0.7014  
Epoch 65/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6165 - acc: 0.6979  
Epoch 66/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6374 - acc: 0.6892  
Epoch 67/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6272 - acc: 0.6927  
Epoch 68/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6249 - acc: 0.6910  
Epoch 69/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6134 - acc: 0.7014  
Epoch 70/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6354 - acc: 0.6753  
Epoch 71/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6250 - acc: 0.6892  
Epoch 72/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6303 - acc: 0.7118  
Epoch 73/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6232 - acc: 0.7014  
Epoch 74/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6190 - acc: 0.6927  
Epoch 75/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6264 - acc: 0.7014  
Epoch 76/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6132 - acc: 0.7049  
Epoch 77/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6078 - acc: 0.7135  
Epoch 78/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6208 - acc: 0.6962  
Epoch 79/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6093 - acc: 0.6944  
Epoch 80/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6160 - acc: 0.7049  
Epoch 81/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6119 - acc: 0.7049  
Epoch 82/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6050 - acc: 0.6997  
Epoch 83/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6051 - acc: 0.6806  
Epoch 84/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6088 - acc: 0.6997  
Epoch 85/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6145 - acc: 0.7049  
Epoch 86/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6015 - acc: 0.7049  
Epoch 87/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6169 - acc: 0.6927  
Epoch 88/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5918 - acc: 0.7031  
Epoch 89/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6060 - acc: 0.7222  
Epoch 90/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5900 - acc: 0.7135  
Epoch 91/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6035 - acc: 0.7031  
Epoch 92/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6259 - acc: 0.6875  
Epoch 93/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5979 - acc: 0.7240  
Epoch 94/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6399 - acc: 0.6597  
Epoch 95/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6146 - acc: 0.7049  
Epoch 96/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6088 - acc: 0.6962  
Epoch 97/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5884 - acc: 0.7326  
Epoch 98/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5898 - acc: 0.7101  
Epoch 99/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5827 - acc: 0.7222  
Epoch 100/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5817 - acc: 0.7170  
Model: "sequential\_15"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 Layer (type) Output Shape Param #   
=================================================================  
 dense\_35 (Dense) (None, 20) 180   
   
 dense\_36 (Dense) (None, 1) 21   
   
=================================================================  
Total params: 201  
Trainable params: 201  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
None  
6/6 [==============================] - 0s 4ms/step - loss: 0.7420 - acc: 0.6927  
[0.7420045733451843, 0.6927083134651184]

Question 3: Normalize the data before feeding the data to the model and check how the normalization change your accuracy (code given below). from sklearn.preprocessing import StandardScaler sc = StandardScaler()

from google.colab import drive  
drive.mount('/content/gdrive')

Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force\_remount=True).

path\_to\_csv = '/content/gdrive/My Drive/breastcancer.csv'

from sklearn.preprocessing import StandardScaler  
sc = StandardScaler()

import keras  
import pandas  
from keras.models import Sequential  
from keras.datasets import mnist  
from keras.layers.core import Dense, Activation  
from sklearn.datasets import load\_breast\_cancer  
from sklearn.model\_selection import train\_test\_split  
import pandas as pd  
import numpy as np  
  
# Load dataset  
  
cancer\_dataset = load\_breast\_cancer()  
X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(dataset[:,0:8], dataset[:,8],  
 test\_size=0.25, random\_state=87)  
  
np.random.seed(155)  
my\_first\_nn = Sequential() # create model  
my\_first\_nn.add(Dense(20, input\_dim=8, activation='relu')) # hidden layer  
my\_first\_nn.add(Dense(1, activation='sigmoid')) # output layer  
my\_first\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])  
my\_first\_nn\_fitted = my\_first\_nn.fit(X\_train, Y\_train, epochs=100,  
 initial\_epoch=0)  
print(my\_first\_nn.summary())  
print(my\_first\_nn.evaluate(X\_test, Y\_test))

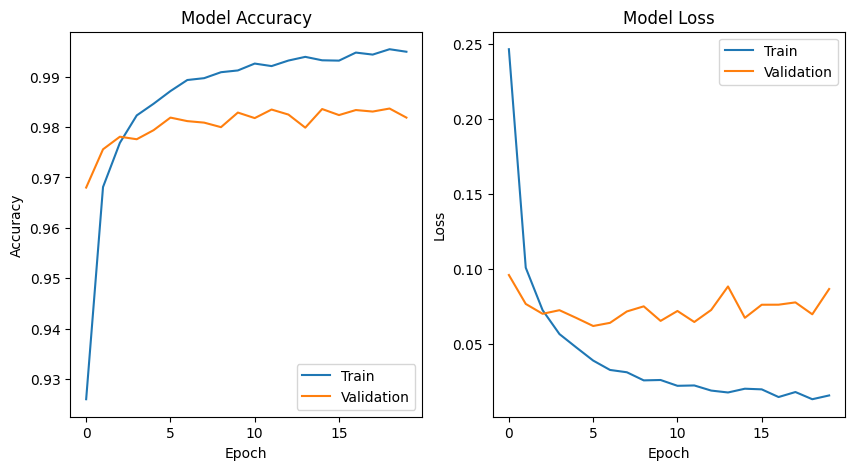
Epoch 1/100  
18/18 [==============================] - 1s 2ms/step - loss: 26.0407 - acc: 0.3385  
Epoch 2/100  
18/18 [==============================] - 0s 2ms/step - loss: 12.8861 - acc: 0.3333  
Epoch 3/100  
18/18 [==============================] - 0s 2ms/step - loss: 5.5346 - acc: 0.4688  
Epoch 4/100  
18/18 [==============================] - 0s 2ms/step - loss: 4.4720 - acc: 0.5868  
Epoch 5/100  
18/18 [==============================] - 0s 2ms/step - loss: 3.7877 - acc: 0.5920  
Epoch 6/100  
18/18 [==============================] - 0s 2ms/step - loss: 3.1827 - acc: 0.6111  
Epoch 7/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.7545 - acc: 0.6024  
Epoch 8/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.4694 - acc: 0.6128  
Epoch 9/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.2956 - acc: 0.6111  
Epoch 10/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.1459 - acc: 0.6094  
Epoch 11/100  
18/18 [==============================] - 0s 2ms/step - loss: 2.0160 - acc: 0.6146  
Epoch 12/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.8903 - acc: 0.6059  
Epoch 13/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.7640 - acc: 0.6319  
Epoch 14/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.6726 - acc: 0.6198  
Epoch 15/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.5617 - acc: 0.6250  
Epoch 16/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.4600 - acc: 0.6302  
Epoch 17/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.3608 - acc: 0.6215  
Epoch 18/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.3382 - acc: 0.6319  
Epoch 19/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.2423 - acc: 0.6233  
Epoch 20/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.1545 - acc: 0.6146  
Epoch 21/100  
18/18 [==============================] - 0s 3ms/step - loss: 1.0918 - acc: 0.6302  
Epoch 22/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.1650 - acc: 0.6146  
Epoch 23/100  
18/18 [==============================] - 0s 2ms/step - loss: 1.0514 - acc: 0.6111  
Epoch 24/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9926 - acc: 0.6372  
Epoch 25/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9474 - acc: 0.6233  
Epoch 26/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9403 - acc: 0.6389  
Epoch 27/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9371 - acc: 0.6250  
Epoch 28/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.9214 - acc: 0.6389  
Epoch 29/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.8998 - acc: 0.6389  
Epoch 30/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.8513 - acc: 0.6476  
Epoch 31/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.8253 - acc: 0.6684  
Epoch 32/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.8450 - acc: 0.6285  
Epoch 33/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.8010 - acc: 0.6545  
Epoch 34/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7838 - acc: 0.6649  
Epoch 35/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7803 - acc: 0.6476  
Epoch 36/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.7480 - acc: 0.6892  
Epoch 37/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7402 - acc: 0.6719  
Epoch 38/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7455 - acc: 0.6667  
Epoch 39/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7339 - acc: 0.6771  
Epoch 40/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7225 - acc: 0.6771  
Epoch 41/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7178 - acc: 0.6684  
Epoch 42/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7276 - acc: 0.6736  
Epoch 43/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7216 - acc: 0.6823  
Epoch 44/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7041 - acc: 0.6562  
Epoch 45/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.7064 - acc: 0.7031  
Epoch 46/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6912 - acc: 0.6944  
Epoch 47/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6972 - acc: 0.6736  
Epoch 48/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7009 - acc: 0.6823  
Epoch 49/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6773 - acc: 0.7031  
Epoch 50/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6905 - acc: 0.6753  
Epoch 51/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6808 - acc: 0.7049  
Epoch 52/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7009 - acc: 0.6788  
Epoch 53/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.7151 - acc: 0.7101  
Epoch 54/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6633 - acc: 0.6979  
Epoch 55/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6601 - acc: 0.7153  
Epoch 56/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6557 - acc: 0.7049  
Epoch 57/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6488 - acc: 0.7031  
Epoch 58/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6495 - acc: 0.6927  
Epoch 59/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6414 - acc: 0.7292  
Epoch 60/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6493 - acc: 0.6858  
Epoch 61/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6446 - acc: 0.7101  
Epoch 62/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6468 - acc: 0.6979  
Epoch 63/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6394 - acc: 0.7222  
Epoch 64/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6322 - acc: 0.6962  
Epoch 65/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6249 - acc: 0.7240  
Epoch 66/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6311 - acc: 0.7188  
Epoch 67/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6324 - acc: 0.7031  
Epoch 68/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6241 - acc: 0.7240  
Epoch 69/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6376 - acc: 0.6997  
Epoch 70/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6167 - acc: 0.7205  
Epoch 71/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6356 - acc: 0.7101  
Epoch 72/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6204 - acc: 0.6944  
Epoch 73/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6344 - acc: 0.7014  
Epoch 74/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6172 - acc: 0.7135  
Epoch 75/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6116 - acc: 0.7205  
Epoch 76/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6105 - acc: 0.7257  
Epoch 77/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6071 - acc: 0.7240  
Epoch 78/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6092 - acc: 0.7326  
Epoch 79/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6242 - acc: 0.6962  
Epoch 80/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.6530 - acc: 0.7153  
Epoch 81/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6392 - acc: 0.6944  
Epoch 82/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6796 - acc: 0.6892  
Epoch 83/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.6148 - acc: 0.6997  
Epoch 84/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5879 - acc: 0.7274  
Epoch 85/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5908 - acc: 0.7274  
Epoch 86/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5816 - acc: 0.7326  
Epoch 87/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5812 - acc: 0.7361  
Epoch 88/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5940 - acc: 0.7153  
Epoch 89/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5820 - acc: 0.7257  
Epoch 90/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5826 - acc: 0.7205  
Epoch 91/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5889 - acc: 0.7326  
Epoch 92/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6194 - acc: 0.7118  
Epoch 93/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.6013 - acc: 0.7240  
Epoch 94/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5794 - acc: 0.7413  
Epoch 95/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5891 - acc: 0.7448  
Epoch 96/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5754 - acc: 0.7170  
Epoch 97/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5840 - acc: 0.7309  
Epoch 98/100  
18/18 [==============================] - 0s 3ms/step - loss: 0.5831 - acc: 0.7222  
Epoch 99/100  
18/18 [==============================] - 0s 4ms/step - loss: 0.5658 - acc: 0.7396  
Epoch 100/100  
18/18 [==============================] - 0s 2ms/step - loss: 0.5649 - acc: 0.7431  
Model: "sequential\_16"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 Layer (type) Output Shape Param #   
=================================================================  
 dense\_37 (Dense) (None, 20) 180   
   
 dense\_38 (Dense) (None, 1) 21   
   
=================================================================  
Total params: 201  
Trainable params: 201  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
None  
6/6 [==============================] - 0s 4ms/step - loss: 0.6311 - acc: 0.6719  
[0.6310913562774658, 0.671875]

Problem 2:

Plot the loss and accuracy for both training data and validation data using the history object in the source code.

import keras  
from keras.datasets import mnist  
from keras.models import Sequential  
from keras.layers import Dense, Dropout  
import matplotlib.pyplot as plt  
  
# load MNIST dataset  
(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  
  
# normalize pixel values to range [0, 1]  
x\_train = x\_train.astype('float32') / 255  
x\_test = x\_test.astype('float32') / 255  
  
# convert class labels to binary class matrices  
num\_classes = 10  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)  
  
# create a simple neural network model  
model = Sequential()  
model.add(Dense(512, activation='relu', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='relu'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
  
model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])  
  
# train the model and record the training history  
history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),  
 epochs=20, batch\_size=128)  
  
# plot the training and validation accuracy and loss curves  
plt.figure(figsize=(10, 5))  
plt.subplot(1, 2, 1)  
plt.plot(history.history['accuracy'])  
plt.plot(history.history['val\_accuracy'])  
plt.title('Model Accuracy')  
plt.ylabel('Accuracy')  
plt.xlabel('Epoch')  
plt.legend(['Train', 'Validation'], loc='lower right')  
  
plt.subplot(1, 2, 2)  
plt.plot(history.history['loss'])  
plt.plot(history.history['val\_loss'])  
plt.title('Model Loss')  
plt.ylabel('Loss')  
plt.xlabel('Epoch')  
plt.legend(['Train', 'Validation'], loc='upper right')  
  
plt.show()

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz  
11490434/11490434 [==============================] - 0s 0us/step  
Epoch 1/20  
469/469 [==============================] - 15s 29ms/step - loss: 0.2464 - accuracy: 0.9259 - val\_loss: 0.0961 - val\_accuracy: 0.9680  
Epoch 2/20  
469/469 [==============================] - 10s 21ms/step - loss: 0.1009 - accuracy: 0.9681 - val\_loss: 0.0768 - val\_accuracy: 0.9756  
Epoch 3/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0727 - accuracy: 0.9769 - val\_loss: 0.0702 - val\_accuracy: 0.9781  
Epoch 4/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0567 - accuracy: 0.9823 - val\_loss: 0.0726 - val\_accuracy: 0.9776  
Epoch 5/20  
469/469 [==============================] - 12s 26ms/step - loss: 0.0478 - accuracy: 0.9847 - val\_loss: 0.0675 - val\_accuracy: 0.9794  
Epoch 6/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0391 - accuracy: 0.9872 - val\_loss: 0.0621 - val\_accuracy: 0.9819  
Epoch 7/20  
469/469 [==============================] - 11s 24ms/step - loss: 0.0329 - accuracy: 0.9894 - val\_loss: 0.0642 - val\_accuracy: 0.9812  
Epoch 8/20  
469/469 [==============================] - 13s 29ms/step - loss: 0.0313 - accuracy: 0.9897 - val\_loss: 0.0718 - val\_accuracy: 0.9809  
Epoch 9/20  
469/469 [==============================] - 12s 26ms/step - loss: 0.0259 - accuracy: 0.9909 - val\_loss: 0.0752 - val\_accuracy: 0.9800  
Epoch 10/20  
469/469 [==============================] - 12s 26ms/step - loss: 0.0262 - accuracy: 0.9913 - val\_loss: 0.0655 - val\_accuracy: 0.9829  
Epoch 11/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0223 - accuracy: 0.9926 - val\_loss: 0.0721 - val\_accuracy: 0.9818  
Epoch 12/20  
469/469 [==============================] - 11s 24ms/step - loss: 0.0225 - accuracy: 0.9921 - val\_loss: 0.0648 - val\_accuracy: 0.9835  
Epoch 13/20  
469/469 [==============================] - 12s 26ms/step - loss: 0.0191 - accuracy: 0.9932 - val\_loss: 0.0728 - val\_accuracy: 0.9825  
Epoch 14/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0178 - accuracy: 0.9940 - val\_loss: 0.0884 - val\_accuracy: 0.9799  
Epoch 15/20  
469/469 [==============================] - 11s 24ms/step - loss: 0.0204 - accuracy: 0.9933 - val\_loss: 0.0676 - val\_accuracy: 0.9836  
Epoch 16/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0199 - accuracy: 0.9932 - val\_loss: 0.0763 - val\_accuracy: 0.9824  
Epoch 17/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0148 - accuracy: 0.9948 - val\_loss: 0.0763 - val\_accuracy: 0.9834  
Epoch 18/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0181 - accuracy: 0.9944 - val\_loss: 0.0778 - val\_accuracy: 0.9831  
Epoch 19/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0134 - accuracy: 0.9955 - val\_loss: 0.0699 - val\_accuracy: 0.9837  
Epoch 20/20  
469/469 [==============================] - 10s 22ms/step - loss: 0.0159 - accuracy: 0.9950 - val\_loss: 0.0868 - val\_accuracy: 0.9819

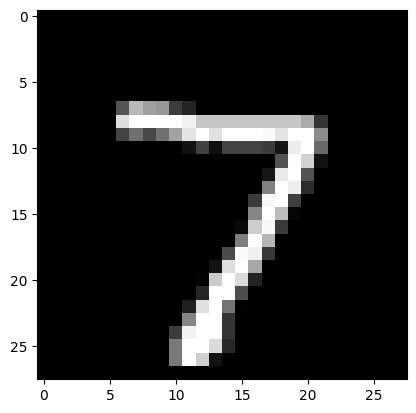


Question 2:

Plot one of the images in the test data, and then do inferencing to check what is the prediction of the model on that single image.

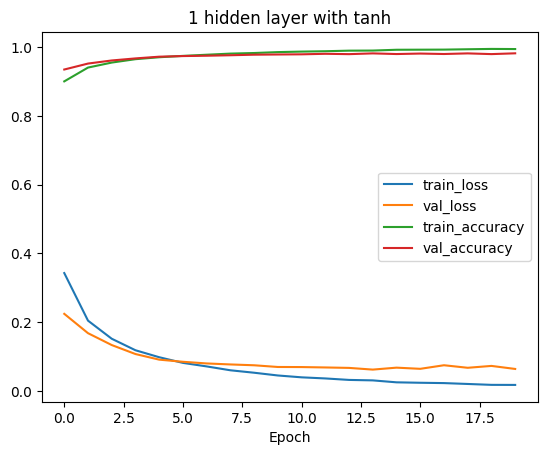
import keras  
from keras.datasets import mnist  
from keras.models import Sequential  
from keras.layers import Dense, Dropout  
import matplotlib.pyplot as plt  
import numpy as np  
  
# load MNIST dataset  
(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  
  
# normalize pixel values to range [0, 1]  
x\_train = x\_train.astype('float32') / 255  
x\_test = x\_test.astype('float32') / 255  
  
# convert class labels to binary class matrices  
num\_classes = 10  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)  
  
# create a simple neural network model  
model = Sequential()  
model.add(Dense(512, activation='relu', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='relu'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
  
model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])  
  
# train the model  
model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),  
 epochs=20, batch\_size=128)  
  
# plot one of the images in the test data  
plt.imshow(x\_test[0], cmap='gray')  
plt.show()  
  
# make a prediction on the image using the trained model  
prediction = model.predict(x\_test[0].reshape(1, -1))  
print('Model prediction:', np.argmax(prediction))

Epoch 1/20  
469/469 [==============================] - 13s 25ms/step - loss: 0.2472 - accuracy: 0.9254 - val\_loss: 0.1070 - val\_accuracy: 0.9646  
Epoch 2/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0992 - accuracy: 0.9696 - val\_loss: 0.0854 - val\_accuracy: 0.9740  
Epoch 3/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0693 - accuracy: 0.9774 - val\_loss: 0.0677 - val\_accuracy: 0.9794  
Epoch 4/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0562 - accuracy: 0.9821 - val\_loss: 0.0560 - val\_accuracy: 0.9827  
Epoch 5/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0455 - accuracy: 0.9857 - val\_loss: 0.0655 - val\_accuracy: 0.9799  
Epoch 6/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0392 - accuracy: 0.9875 - val\_loss: 0.0618 - val\_accuracy: 0.9816  
Epoch 7/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0346 - accuracy: 0.9887 - val\_loss: 0.0660 - val\_accuracy: 0.9827  
Epoch 8/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0291 - accuracy: 0.9902 - val\_loss: 0.0607 - val\_accuracy: 0.9826  
Epoch 9/20  
469/469 [==============================] - 13s 27ms/step - loss: 0.0266 - accuracy: 0.9910 - val\_loss: 0.0733 - val\_accuracy: 0.9811  
Epoch 10/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0273 - accuracy: 0.9908 - val\_loss: 0.0760 - val\_accuracy: 0.9819  
Epoch 11/20  
469/469 [==============================] - 11s 24ms/step - loss: 0.0240 - accuracy: 0.9919 - val\_loss: 0.0748 - val\_accuracy: 0.9814  
Epoch 12/20  
469/469 [==============================] - 10s 21ms/step - loss: 0.0222 - accuracy: 0.9929 - val\_loss: 0.0838 - val\_accuracy: 0.9791  
Epoch 13/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0185 - accuracy: 0.9937 - val\_loss: 0.0716 - val\_accuracy: 0.9821  
Epoch 14/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0178 - accuracy: 0.9940 - val\_loss: 0.0875 - val\_accuracy: 0.9812  
Epoch 15/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0216 - accuracy: 0.9928 - val\_loss: 0.0667 - val\_accuracy: 0.9829  
Epoch 16/20  
469/469 [==============================] - 11s 23ms/step - loss: 0.0146 - accuracy: 0.9950 - val\_loss: 0.0796 - val\_accuracy: 0.9836  
Epoch 17/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0158 - accuracy: 0.9951 - val\_loss: 0.0758 - val\_accuracy: 0.9842  
Epoch 18/20  
469/469 [==============================] - 12s 25ms/step - loss: 0.0164 - accuracy: 0.9943 - val\_loss: 0.0817 - val\_accuracy: 0.9817  
Epoch 19/20  
469/469 [==============================] - 12s 26ms/step - loss: 0.0162 - accuracy: 0.9944 - val\_loss: 0.0766 - val\_accuracy: 0.9835  
Epoch 20/20  
469/469 [==============================] - 19s 39ms/step - loss: 0.0178 - accuracy: 0.9942 - val\_loss: 0.0749 - val\_accuracy: 0.9855

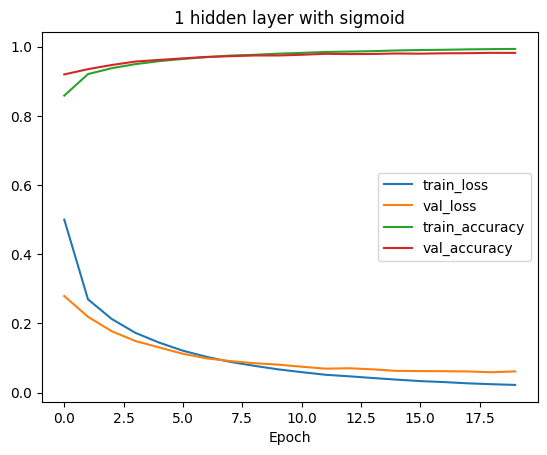


1/1 [==============================] - 0s 63ms/step  
Model prediction: 7

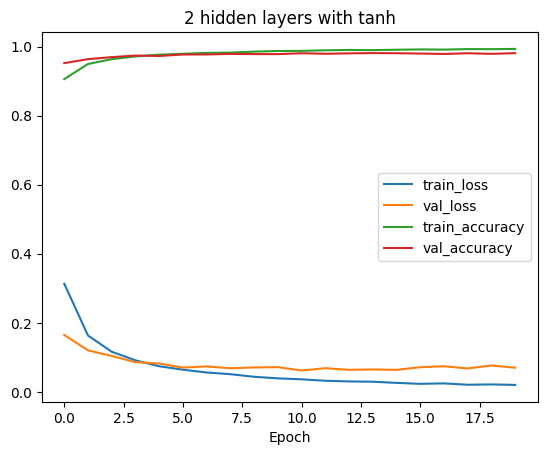
import keras  
from keras.datasets import mnist  
from keras.models import Sequential  
from keras.layers import Dense, Dropout  
import matplotlib.pyplot as plt  
import numpy as np  
  
# load MNIST dataset  
(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  
  
# normalize pixel values to range [0, 1]  
x\_train = x\_train.astype('float32') / 255  
x\_test = x\_test.astype('float32') / 255  
  
# convert class labels to binary class matrices  
num\_classes = 10  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)  
  
# create a list of models to train  
models = []  
  
# model with 1 hidden layer and tanh activation  
model = Sequential()  
model.add(Dense(512, activation='tanh', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('1 hidden layer with tanh', model))  
  
# model with 1 hidden layer and sigmoid activation  
model = Sequential()  
model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('1 hidden layer with sigmoid', model))  
  
# model with 2 hidden layers and tanh activation  
model = Sequential()  
model.add(Dense(512, activation='tanh', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='tanh'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('2 hidden layers with tanh', model))  
  
# model with 2 hidden layers and sigmoid activation  
model = Sequential()  
model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='sigmoid'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('2 hidden layers with sigmoid', model))  
  
# train each model and plot loss and accuracy curves  
for name, model in models:  
 model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])  
 history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),  
 epochs=20, batch\_size=128, verbose=0)  
 # plot loss and accuracy curves  
 plt.plot(history.history['loss'], label='train\_loss')  
 plt.plot(history.history['val\_loss'], label='val\_loss')  
 plt.plot(history.history['accuracy'], label='train\_accuracy')  
 plt.plot(history.history['val\_accuracy'], label='val\_accuracy')  
 plt.title(name)  
 plt.xlabel('Epoch')  
 plt.legend()  
 plt.show()  
  
 # evaluate the model on test data  
 loss, accuracy = model.evaluate(x\_test.reshape(-1, 784), y\_test, verbose=0)  
 print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))



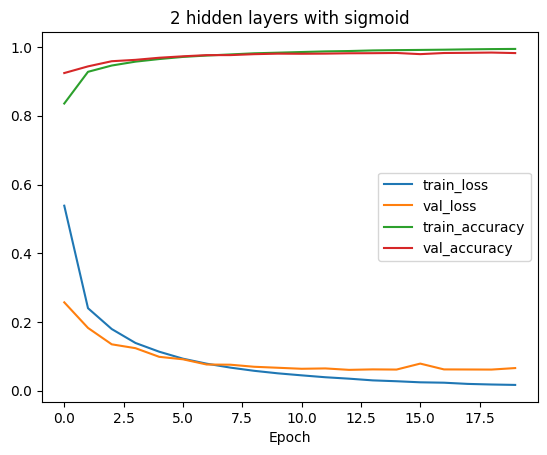
1 hidden layer with tanh - Test loss: 0.0640, Test accuracy: 0.9817



1 hidden layer with sigmoid - Test loss: 0.0610, Test accuracy: 0.9822

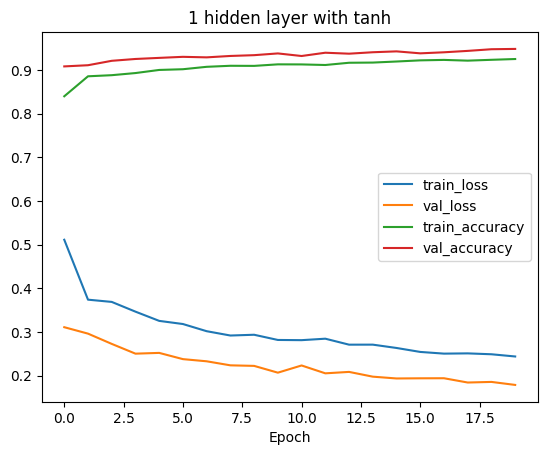


2 hidden layers with tanh - Test loss: 0.0700, Test accuracy: 0.9810

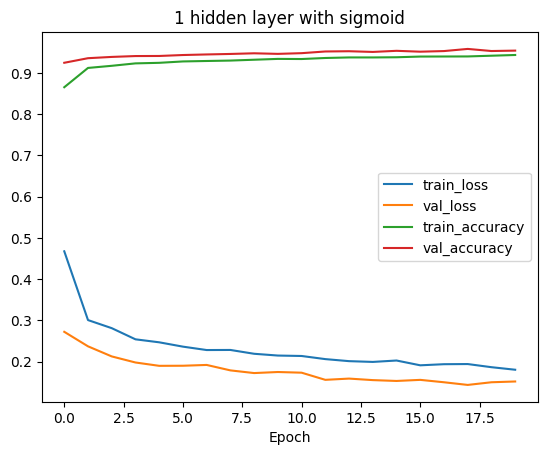


2 hidden layers with sigmoid - Test loss: 0.0659, Test accuracy: 0.9824

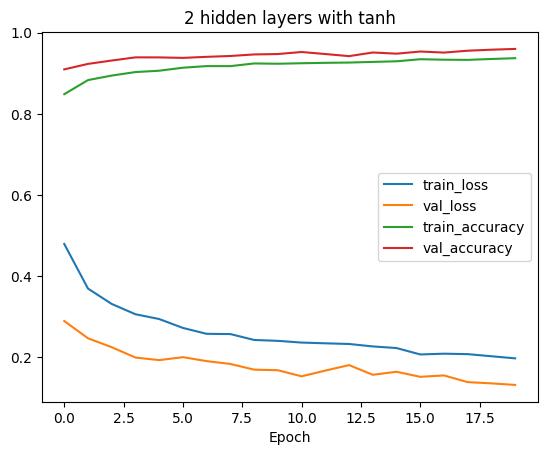
import keras  
from keras.datasets import mnist  
from keras.models import Sequential  
from keras.layers import Dense, Dropout  
import matplotlib.pyplot as plt  
import numpy as np  
  
# load MNIST dataset  
(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  
  
# convert class labels to binary class matrices  
num\_classes = 10  
y\_train = keras.utils.to\_categorical(y\_train, num\_classes)  
y\_test = keras.utils.to\_categorical(y\_test, num\_classes)  
  
# create a list of models to train  
models = []  
  
# model with 1 hidden layer and tanh activation  
model = Sequential()  
model.add(Dense(512, activation='tanh', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('1 hidden layer with tanh', model))  
  
# model with 1 hidden layer and sigmoid activation  
model = Sequential()  
model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('1 hidden layer with sigmoid', model))  
  
# model with 2 hidden layers and tanh activation  
model = Sequential()  
model.add(Dense(512, activation='tanh', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='tanh'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('2 hidden layers with tanh', model))  
  
# model with 2 hidden layers and sigmoid activation  
model = Sequential()  
model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))  
model.add(Dropout(0.2))  
model.add(Dense(512, activation='sigmoid'))  
model.add(Dropout(0.2))  
model.add(Dense(num\_classes, activation='softmax'))  
models.append(('2 hidden layers with sigmoid', model))  
  
# train each model and plot loss and accuracy curves  
for name, model in models:  
 model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])  
 history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),  
 epochs=20, batch\_size=128, verbose=0)  
 # plot loss and accuracy curves  
 plt.plot(history.history['loss'], label='train\_loss')  
 plt.plot(history.history['val\_loss'], label='val\_loss')  
 plt.plot(history.history['accuracy'], label='train\_accuracy')  
 plt.plot(history.history['val\_accuracy'], label='val\_accuracy')  
 plt.title(name)  
 plt.xlabel('Epoch')  
 plt.legend()  
 plt.show()  
  
 # evaluate the model on test data  
 loss, accuracy = model.evaluate(x\_test.reshape(-1, 784), y\_test, verbose=0)  
 print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))



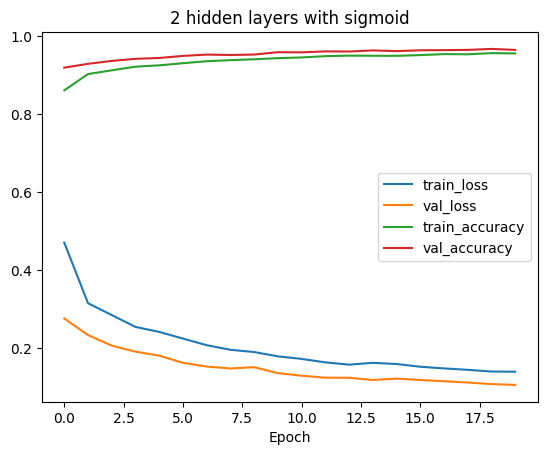
1 hidden layer with tanh - Test loss: 0.1791, Test accuracy: 0.9479



1 hidden layer with sigmoid - Test loss: 0.1518, Test accuracy: 0.9544



2 hidden layers with tanh - Test loss: 0.1317, Test accuracy: 0.9600



2 hidden layers with sigmoid - Test loss: 0.1061, Test accuracy: 0.9645