{  
 "nbformat": *4*,  
 "nbformat\_minor": *0*,  
 "metadata": {  
 "colab": {  
 "provenance": []  
 },  
 "kernelspec": {  
 "name": "python3",  
 "display\_name": "Python 3"  
 },  
 "language\_info": {  
 "name": "python"  
 }  
 },  
 "cells": [  
 {  
 "cell\_type": "code",  
 "source": [  
 "#read the data*\n*",  
 "data = pd.read\_csv('sample\_data/diabetes.csv')"  
 ],  
 "metadata": {  
 "id": "YU1Lh1NtmBuL"  
 },  
 "execution\_count": *58*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "path\_to\_csv = 'sample\_data/diabetes.csv'"  
 ],  
 "metadata": {  
 "id": "2dPHGcJ1mM2j"  
 },  
 "execution\_count": *59*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "import pandas*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers.core import Dense, Activation*\n*",  
 "*\n*",  
 "# load dataset*\n*",  
 "from sklearn.model\_selection import train\_test\_split*\n*",  
 "import pandas as pd*\n*",  
 "import numpy as np*\n*",  
 "*\n*",  
 "dataset = pd.read\_csv(path\_to\_csv, header=None).values*\n*",  
 "*\n*",  
 "X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(dataset[:,0:8], dataset[:,8],*\n*",  
 " test\_size=0.25, random\_state=87)*\n*",  
 "np.random.seed(155)*\n*",  
 "my\_first\_nn = Sequential() # create model*\n*",  
 "my\_first\_nn.add(Dense(20, input\_dim=8, activation='relu')) # hidden layer*\n*",  
 "my\_first\_nn.add(Dense(4, activation='relu')) # hidden layer*\n*",  
 "my\_first\_nn.add(Dense(1, activation='sigmoid')) # output layer*\n*",  
 "my\_first\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])*\n*",  
 "my\_first\_nn\_fitted = my\_first\_nn.fit(X\_train, Y\_train, epochs=100,*\n*",  
 " initial\_epoch=0)*\n*",  
 "print(my\_first\_nn.summary())*\n*",  
 "print(my\_first\_nn.evaluate(X\_test, Y\_test))"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/"  
 },  
 "id": "bx95Fv4bmRdH",  
 "outputId": "176c1d0a-2cb4-4361-f2b3-764cfc4290f7"  
 },  
 "execution\_count": *63*,  
 "outputs": [  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "Epoch 1/100*\n*",  
 "18/18 [==============================] - 1s 2ms/step - loss: 18.2141 - acc: 0.3385*\n*",  
 "Epoch 2/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 8.1899 - acc: 0.3438*\n*",  
 "Epoch 3/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 1.7616 - acc: 0.3924*\n*",  
 "Epoch 4/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.8124 - acc: 0.5278*\n*",  
 "Epoch 5/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.7466 - acc: 0.5972*\n*",  
 "Epoch 6/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.7242 - acc: 0.6181*\n*",  
 "Epoch 7/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.7203 - acc: 0.6319*\n*",  
 "Epoch 8/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.7132 - acc: 0.6458*\n*",  
 "Epoch 9/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.7066 - acc: 0.6458*\n*",  
 "Epoch 10/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.7044 - acc: 0.6441*\n*",  
 "Epoch 11/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.7018 - acc: 0.6545*\n*",  
 "Epoch 12/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6989 - acc: 0.6545*\n*",  
 "Epoch 13/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.7013 - acc: 0.6580*\n*",  
 "Epoch 14/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6929 - acc: 0.6493*\n*",  
 "Epoch 15/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6911 - acc: 0.6528*\n*",  
 "Epoch 16/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6882 - acc: 0.6545*\n*",  
 "Epoch 17/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6849 - acc: 0.6528*\n*",  
 "Epoch 18/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6877 - acc: 0.6545*\n*",  
 "Epoch 19/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6785 - acc: 0.6615*\n*",  
 "Epoch 20/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6775 - acc: 0.6649*\n*",  
 "Epoch 21/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6738 - acc: 0.6615*\n*",  
 "Epoch 22/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6761 - acc: 0.6632*\n*",  
 "Epoch 23/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6763 - acc: 0.6597*\n*",  
 "Epoch 24/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6713 - acc: 0.6632*\n*",  
 "Epoch 25/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6719 - acc: 0.6632*\n*",  
 "Epoch 26/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6687 - acc: 0.6632*\n*",  
 "Epoch 27/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6654 - acc: 0.6649*\n*",  
 "Epoch 28/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6669 - acc: 0.6684*\n*",  
 "Epoch 29/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6644 - acc: 0.6597*\n*",  
 "Epoch 30/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6656 - acc: 0.6684*\n*",  
 "Epoch 31/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6611 - acc: 0.6632*\n*",  
 "Epoch 32/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6615 - acc: 0.6632*\n*",  
 "Epoch 33/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6592 - acc: 0.6684*\n*",  
 "Epoch 34/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6585 - acc: 0.6632*\n*",  
 "Epoch 35/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6564 - acc: 0.6701*\n*",  
 "Epoch 36/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6569 - acc: 0.6580*\n*",  
 "Epoch 37/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6594 - acc: 0.6667*\n*",  
 "Epoch 38/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6690 - acc: 0.6649*\n*",  
 "Epoch 39/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6554 - acc: 0.6701*\n*",  
 "Epoch 40/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6519 - acc: 0.6684*\n*",  
 "Epoch 41/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6506 - acc: 0.6667*\n*",  
 "Epoch 42/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6493 - acc: 0.6701*\n*",  
 "Epoch 43/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6495 - acc: 0.6719*\n*",  
 "Epoch 44/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6639 - acc: 0.6649*\n*",  
 "Epoch 45/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6552 - acc: 0.6736*\n*",  
 "Epoch 46/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6501 - acc: 0.6719*\n*",  
 "Epoch 47/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6461 - acc: 0.6736*\n*",  
 "Epoch 48/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6469 - acc: 0.6667*\n*",  
 "Epoch 49/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6464 - acc: 0.6719*\n*",  
 "Epoch 50/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6409 - acc: 0.6736*\n*",  
 "Epoch 51/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6433 - acc: 0.6736*\n*",  
 "Epoch 52/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6428 - acc: 0.6719*\n*",  
 "Epoch 53/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6420 - acc: 0.6736*\n*",  
 "Epoch 54/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6409 - acc: 0.6719*\n*",  
 "Epoch 55/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6403 - acc: 0.6719*\n*",  
 "Epoch 56/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6408 - acc: 0.6719*\n*",  
 "Epoch 57/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6408 - acc: 0.6684*\n*",  
 "Epoch 58/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6404 - acc: 0.6719*\n*",  
 "Epoch 59/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6404 - acc: 0.6701*\n*",  
 "Epoch 60/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6390 - acc: 0.6736*\n*",  
 "Epoch 61/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6389 - acc: 0.6753*\n*",  
 "Epoch 62/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6719*\n*",  
 "Epoch 63/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6382 - acc: 0.6771*\n*",  
 "Epoch 64/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6736*\n*",  
 "Epoch 65/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6363 - acc: 0.6771*\n*",  
 "Epoch 66/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6374 - acc: 0.6753*\n*",  
 "Epoch 67/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6361 - acc: 0.6736*\n*",  
 "Epoch 68/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6359 - acc: 0.6719*\n*",  
 "Epoch 69/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6351 - acc: 0.6701*\n*",  
 "Epoch 70/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6340 - acc: 0.6788*\n*",  
 "Epoch 71/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6333 - acc: 0.6771*\n*",  
 "Epoch 72/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6397 - acc: 0.6701*\n*",  
 "Epoch 73/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6341 - acc: 0.6649*\n*",  
 "Epoch 74/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6338 - acc: 0.6771*\n*",  
 "Epoch 75/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6360 - acc: 0.6753*\n*",  
 "Epoch 76/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6339 - acc: 0.6753*\n*",  
 "Epoch 77/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6329 - acc: 0.6788*\n*",  
 "Epoch 78/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6326 - acc: 0.6771*\n*",  
 "Epoch 79/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6370 - acc: 0.6771*\n*",  
 "Epoch 80/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6319 - acc: 0.6771*\n*",  
 "Epoch 81/100*\n*",  
 "18/18 [==============================] - 0s 2ms/step - loss: 0.6310 - acc: 0.6771*\n*",  
 "Epoch 82/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6299 - acc: 0.6806*\n*",  
 "Epoch 83/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6293 - acc: 0.6823*\n*",  
 "Epoch 84/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6298 - acc: 0.6753*\n*",  
 "Epoch 85/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6306 - acc: 0.6753*\n*",  
 "Epoch 86/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6328 - acc: 0.6719*\n*",  
 "Epoch 87/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6326 - acc: 0.6771*\n*",  
 "Epoch 88/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6301 - acc: 0.6771*\n*",  
 "Epoch 89/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6287 - acc: 0.6788*\n*",  
 "Epoch 90/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6293 - acc: 0.6719*\n*",  
 "Epoch 91/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6261 - acc: 0.6771*\n*",  
 "Epoch 92/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6234 - acc: 0.6823*\n*",  
 "Epoch 93/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6268 - acc: 0.6753*\n*",  
 "Epoch 94/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6275 - acc: 0.6823*\n*",  
 "Epoch 95/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6320 - acc: 0.6806*\n*",  
 "Epoch 96/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6425 - acc: 0.6771*\n*",  
 "Epoch 97/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6823*\n*",  
 "Epoch 98/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6220 - acc: 0.6806*\n*",  
 "Epoch 99/100*\n*",  
 "18/18 [==============================] - 0s 4ms/step - loss: 0.6210 - acc: 0.6858*\n*",  
 "Epoch 100/100*\n*",  
 "18/18 [==============================] - 0s 3ms/step - loss: 0.6212 - acc: 0.6823*\n*",  
 "Model: *\"*sequential\_38*\"\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 " Layer (type) Output Shape Param # *\n*",  
 "=================================================================*\n*",  
 " dense\_89 (Dense) (None, 20) 180 *\n*",  
 " *\n*",  
 " dense\_90 (Dense) (None, 4) 84 *\n*",  
 " *\n*",  
 " dense\_91 (Dense) (None, 1) 5 *\n*",  
 " *\n*",  
 "=================================================================*\n*",  
 "Total params: 269*\n*",  
 "Trainable params: 269*\n*",  
 "Non-trainable params: 0*\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 "None*\n*",  
 "6/6 [==============================] - 0s 4ms/step - loss: 0.7119 - acc: 0.5833*\n*",  
 "[0.7118666172027588, 0.5833333134651184]*\n*"  
 ]  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "#read the data*\n*",  
 "data = pd.read\_csv('sample\_data/breastcancer.csv')"  
 ],  
 "metadata": {  
 "id": "kjSawgmgRW5f"  
 },  
 "execution\_count": *72*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "path\_to\_csv = 'sample\_data/breastcancer.csv'"  
 ],  
 "metadata": {  
 "id": "HItyHrNXRu5T"  
 },  
 "execution\_count": *73*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "import pandas as pd*\n*",  
 "import numpy as np*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers.core import Dense, Activation*\n*",  
 "from sklearn.datasets import load\_breast\_cancer*\n*",  
 "from sklearn.model\_selection import train\_test\_split*\n*",  
 "*\n*",  
 "# load dataset*\n*",  
 "cancer\_data = load\_breast\_cancer()*\n*",  
 "X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(cancer\_data.data, cancer\_data.target,*\n*",  
 " test\_size=0.25, random\_state=87)*\n*",  
 "np.random.seed(155)*\n*",  
 "my\_nn = Sequential() # create model*\n*",  
 "my\_nn.add(Dense(20, input\_dim=30, activation='relu')) # hidden layer 1*\n*",  
 "my\_nn.add(Dense(1, activation='sigmoid')) # output layer*\n*",  
 "my\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])*\n*",  
 "my\_nn\_fitted = my\_nn.fit(X\_train, Y\_train, epochs=100,*\n*",  
 " initial\_epoch=0)*\n*",  
 "print(my\_nn.summary())*\n*",  
 "print(my\_nn.evaluate(X\_test, Y\_test))*\n*"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/"  
 },  
 "id": "xOark8UsgRXN",  
 "outputId": "9cc1d556-75ea-41f4-a61b-2c20dff81ee3"  
 },  
 "execution\_count": *75*,  
 "outputs": [  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "Epoch 1/100*\n*",  
 "14/14 [==============================] - 1s 5ms/step - loss: 67.9584 - acc: 0.3803*\n*",  
 "Epoch 2/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 20.8848 - acc: 0.3897*\n*",  
 "Epoch 3/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 5.6956 - acc: 0.6901*\n*",  
 "Epoch 4/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 1.8838 - acc: 0.6643*\n*",  
 "Epoch 5/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 1.0273 - acc: 0.8732*\n*",  
 "Epoch 6/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.7197 - acc: 0.8498*\n*",  
 "Epoch 7/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.6906 - acc: 0.8920*\n*",  
 "Epoch 8/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.6208 - acc: 0.8685*\n*",  
 "Epoch 9/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.6028 - acc: 0.8803*\n*",  
 "Epoch 10/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.6171 - acc: 0.8709*\n*",  
 "Epoch 11/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.5705 - acc: 0.8732*\n*",  
 "Epoch 12/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.5579 - acc: 0.8967*\n*",  
 "Epoch 13/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.5383 - acc: 0.8615*\n*",  
 "Epoch 14/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4950 - acc: 0.9038*\n*",  
 "Epoch 15/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.4360 - acc: 0.8826*\n*",  
 "Epoch 16/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.4114 - acc: 0.8967*\n*",  
 "Epoch 17/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.3769 - acc: 0.8897*\n*",  
 "Epoch 18/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.3311 - acc: 0.9038*\n*",  
 "Epoch 19/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.3080 - acc: 0.9085*\n*",  
 "Epoch 20/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.3078 - acc: 0.9061*\n*",  
 "Epoch 21/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2686 - acc: 0.9085*\n*",  
 "Epoch 22/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2805 - acc: 0.9038*\n*",  
 "Epoch 23/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2499 - acc: 0.9085*\n*",  
 "Epoch 24/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.2284 - acc: 0.9202*\n*",  
 "Epoch 25/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2561 - acc: 0.9178*\n*",  
 "Epoch 26/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2419 - acc: 0.9155*\n*",  
 "Epoch 27/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2632 - acc: 0.9155*\n*",  
 "Epoch 28/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2283 - acc: 0.9225*\n*",  
 "Epoch 29/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2308 - acc: 0.9155*\n*",  
 "Epoch 30/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.2116 - acc: 0.9272*\n*",  
 "Epoch 31/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.2453 - acc: 0.9249*\n*",  
 "Epoch 32/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.2582 - acc: 0.9249*\n*",  
 "Epoch 33/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2564 - acc: 0.9108*\n*",  
 "Epoch 34/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2345 - acc: 0.9249*\n*",  
 "Epoch 35/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2230 - acc: 0.9296*\n*",  
 "Epoch 36/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2098 - acc: 0.9296*\n*",  
 "Epoch 37/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1961 - acc: 0.9225*\n*",  
 "Epoch 38/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2062 - acc: 0.9296*\n*",  
 "Epoch 39/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2405 - acc: 0.9202*\n*",  
 "Epoch 40/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.1920 - acc: 0.9319*\n*",  
 "Epoch 41/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2080 - acc: 0.9272*\n*",  
 "Epoch 42/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2147 - acc: 0.9319*\n*",  
 "Epoch 43/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.1979 - acc: 0.9249*\n*",  
 "Epoch 44/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1910 - acc: 0.9249*\n*",  
 "Epoch 45/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2231 - acc: 0.9249*\n*",  
 "Epoch 46/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.1792 - acc: 0.9249*\n*",  
 "Epoch 47/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.1798 - acc: 0.9296*\n*",  
 "Epoch 48/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1899 - acc: 0.9272*\n*",  
 "Epoch 49/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2048 - acc: 0.9272*\n*",  
 "Epoch 50/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.1755 - acc: 0.9319*\n*",  
 "Epoch 51/100*\n*",  
 "14/14 [==============================] - 0s 5ms/step - loss: 0.1809 - acc: 0.9319*\n*",  
 "Epoch 52/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2014 - acc: 0.9225*\n*",  
 "Epoch 53/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2043 - acc: 0.9225*\n*",  
 "Epoch 54/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.2078 - acc: 0.9131*\n*",  
 "Epoch 55/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.1916 - acc: 0.9319*\n*",  
 "Epoch 56/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1831 - acc: 0.9319*\n*",  
 "Epoch 57/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.2104 - acc: 0.9202*\n*",  
 "Epoch 58/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.3084 - acc: 0.8897*\n*",  
 "Epoch 59/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1922 - acc: 0.9272*\n*",  
 "Epoch 60/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1683 - acc: 0.9366*\n*",  
 "Epoch 61/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1779 - acc: 0.9343*\n*",  
 "Epoch 62/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1631 - acc: 0.9319*\n*",  
 "Epoch 63/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1681 - acc: 0.9296*\n*",  
 "Epoch 64/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1728 - acc: 0.9319*\n*",  
 "Epoch 65/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2012 - acc: 0.9319*\n*",  
 "Epoch 66/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1632 - acc: 0.9319*\n*",  
 "Epoch 67/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1814 - acc: 0.9319*\n*",  
 "Epoch 68/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1889 - acc: 0.9319*\n*",  
 "Epoch 69/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1598 - acc: 0.9413*\n*",  
 "Epoch 70/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1856 - acc: 0.9343*\n*",  
 "Epoch 71/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1761 - acc: 0.9343*\n*",  
 "Epoch 72/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1687 - acc: 0.9343*\n*",  
 "Epoch 73/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1785 - acc: 0.9319*\n*",  
 "Epoch 74/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1957 - acc: 0.9272*\n*",  
 "Epoch 75/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1626 - acc: 0.9366*\n*",  
 "Epoch 76/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1751 - acc: 0.9390*\n*",  
 "Epoch 77/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1657 - acc: 0.9296*\n*",  
 "Epoch 78/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2212 - acc: 0.9272*\n*",  
 "Epoch 79/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1976 - acc: 0.9249*\n*",  
 "Epoch 80/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2479 - acc: 0.9108*\n*",  
 "Epoch 81/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2057 - acc: 0.9366*\n*",  
 "Epoch 82/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2357 - acc: 0.9343*\n*",  
 "Epoch 83/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1895 - acc: 0.9319*\n*",  
 "Epoch 84/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1678 - acc: 0.9272*\n*",  
 "Epoch 85/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1747 - acc: 0.9390*\n*",  
 "Epoch 86/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.2110 - acc: 0.9296*\n*",  
 "Epoch 87/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1799 - acc: 0.9249*\n*",  
 "Epoch 88/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1872 - acc: 0.9343*\n*",  
 "Epoch 89/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1507 - acc: 0.9343*\n*",  
 "Epoch 90/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1810 - acc: 0.9319*\n*",  
 "Epoch 91/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.1508 - acc: 0.9319*\n*",  
 "Epoch 92/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.2480 - acc: 0.9225*\n*",  
 "Epoch 93/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2020 - acc: 0.9343*\n*",  
 "Epoch 94/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1698 - acc: 0.9343*\n*",  
 "Epoch 95/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1509 - acc: 0.9390*\n*",  
 "Epoch 96/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1522 - acc: 0.9390*\n*",  
 "Epoch 97/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1466 - acc: 0.9343*\n*",  
 "Epoch 98/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1683 - acc: 0.9319*\n*",  
 "Epoch 99/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.2092 - acc: 0.9178*\n*",  
 "Epoch 100/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.1453 - acc: 0.9484*\n*",  
 "Model: *\"*sequential\_42*\"\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 " Layer (type) Output Shape Param # *\n*",  
 "=================================================================*\n*",  
 " dense\_98 (Dense) (None, 20) 620 *\n*",  
 " *\n*",  
 " dense\_99 (Dense) (None, 1) 21 *\n*",  
 " *\n*",  
 "=================================================================*\n*",  
 "Total params: 641*\n*",  
 "Trainable params: 641*\n*",  
 "Non-trainable params: 0*\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 "None*\n*",  
 "5/5 [==============================] - 0s 4ms/step - loss: 0.3893 - acc: 0.8881*\n*",  
 "[0.3892970681190491, 0.8881118893623352]*\n*"  
 ]  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "#read the data*\n*",  
 "data = pd.read\_csv('sample\_data/breastcancer.csv')"  
 ],  
 "metadata": {  
 "id": "QWlp2coRo-bs"  
 },  
 "execution\_count": *76*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "path\_to\_csv = 'sample\_data/breastcancer.csv'"  
 ],  
 "metadata": {  
 "id": "dQSEk77TpDuM"  
 },  
 "execution\_count": *77*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "from sklearn.preprocessing import StandardScaler*\n*",  
 "sc = StandardScaler()"  
 ],  
 "metadata": {  
 "id": "t6Ohhor-pItF"  
 },  
 "execution\_count": *81*,  
 "outputs": []  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "import pandas as pd*\n*",  
 "import numpy as np*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers.core import Dense, Activation*\n*",  
 "from sklearn.datasets import load\_breast\_cancer*\n*",  
 "from sklearn.model\_selection import train\_test\_split*\n*",  
 "*\n*",  
 "# load dataset*\n*",  
 "cancer\_data = load\_breast\_cancer()*\n*",  
 "X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(cancer\_data.data, cancer\_data.target,*\n*",  
 " test\_size=0.25, random\_state=87)*\n*",  
 "np.random.seed(155)*\n*",  
 "my\_nn = Sequential() # create model*\n*",  
 "my\_nn.add(Dense(20, input\_dim=30, activation='relu')) # hidden layer 1*\n*",  
 "my\_nn.add(Dense(1, activation='sigmoid')) # output layer*\n*",  
 "my\_nn.compile(loss='binary\_crossentropy', optimizer='adam', metrics=['acc'])*\n*",  
 "my\_nn\_fitted = my\_nn.fit(X\_train, Y\_train, epochs=100,*\n*",  
 " initial\_epoch=0)*\n*",  
 "print(my\_nn.summary())*\n*",  
 "print(my\_nn.evaluate(X\_test, Y\_test))*\n*"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/"  
 },  
 "id": "H\_58MlScpTMm",  
 "outputId": "82f10e27-3e7d-4abf-fb30-524c7ddc7cd2"  
 },  
 "execution\_count": *82*,  
 "outputs": [  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "Epoch 1/100*\n*",  
 "14/14 [==============================] - 1s 2ms/step - loss: 173.1653 - acc: 0.6197*\n*",  
 "Epoch 2/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 98.1999 - acc: 0.6197*\n*",  
 "Epoch 3/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 25.2683 - acc: 0.6174*\n*",  
 "Epoch 4/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 11.1987 - acc: 0.4061*\n*",  
 "Epoch 5/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 4.9497 - acc: 0.7324*\n*",  
 "Epoch 6/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 4.4129 - acc: 0.7606*\n*",  
 "Epoch 7/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 4.2134 - acc: 0.6808*\n*",  
 "Epoch 8/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 3.7228 - acc: 0.7746*\n*",  
 "Epoch 9/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 3.4424 - acc: 0.7559*\n*",  
 "Epoch 10/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 3.2638 - acc: 0.7770*\n*",  
 "Epoch 11/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 3.0410 - acc: 0.7840*\n*",  
 "Epoch 12/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 2.8859 - acc: 0.8239*\n*",  
 "Epoch 13/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 2.7474 - acc: 0.7911*\n*",  
 "Epoch 14/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 2.7161 - acc: 0.8263*\n*",  
 "Epoch 15/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 2.4708 - acc: 0.8310*\n*",  
 "Epoch 16/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 2.3567 - acc: 0.8568*\n*",  
 "Epoch 17/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 2.2481 - acc: 0.8592*\n*",  
 "Epoch 18/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 2.1773 - acc: 0.8592*\n*",  
 "Epoch 19/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 2.0691 - acc: 0.8685*\n*",  
 "Epoch 20/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 2.1096 - acc: 0.8169*\n*",  
 "Epoch 21/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 2.1030 - acc: 0.9108*\n*",  
 "Epoch 22/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.8225 - acc: 0.8709*\n*",  
 "Epoch 23/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.7438 - acc: 0.8545*\n*",  
 "Epoch 24/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.7889 - acc: 0.8756*\n*",  
 "Epoch 25/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.6852 - acc: 0.8474*\n*",  
 "Epoch 26/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.5249 - acc: 0.9014*\n*",  
 "Epoch 27/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.5215 - acc: 0.8615*\n*",  
 "Epoch 28/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 1.4493 - acc: 0.8897*\n*",  
 "Epoch 29/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.4228 - acc: 0.8545*\n*",  
 "Epoch 30/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.3600 - acc: 0.8967*\n*",  
 "Epoch 31/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.2938 - acc: 0.8803*\n*",  
 "Epoch 32/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.2588 - acc: 0.9131*\n*",  
 "Epoch 33/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.2027 - acc: 0.8615*\n*",  
 "Epoch 34/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 1.2380 - acc: 0.9155*\n*",  
 "Epoch 35/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.1510 - acc: 0.8756*\n*",  
 "Epoch 36/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.0808 - acc: 0.9038*\n*",  
 "Epoch 37/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 1.0994 - acc: 0.8779*\n*",  
 "Epoch 38/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 1.0331 - acc: 0.8897*\n*",  
 "Epoch 39/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.9756 - acc: 0.9085*\n*",  
 "Epoch 40/100*\n*",  
 "14/14 [==============================] - 0s 4ms/step - loss: 0.9911 - acc: 0.8967*\n*",  
 "Epoch 41/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.9283 - acc: 0.8779*\n*",  
 "Epoch 42/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.9483 - acc: 0.9108*\n*",  
 "Epoch 43/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.8765 - acc: 0.8967*\n*",  
 "Epoch 44/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.8945 - acc: 0.8803*\n*",  
 "Epoch 45/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.8301 - acc: 0.9038*\n*",  
 "Epoch 46/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.8492 - acc: 0.9014*\n*",  
 "Epoch 47/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.9405 - acc: 0.8779*\n*",  
 "Epoch 48/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.8971 - acc: 0.9131*\n*",  
 "Epoch 49/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7844 - acc: 0.8850*\n*",  
 "Epoch 50/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7745 - acc: 0.9108*\n*",  
 "Epoch 51/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.7524 - acc: 0.8944*\n*",  
 "Epoch 52/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 1.0062 - acc: 0.9249*\n*",  
 "Epoch 53/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.8134 - acc: 0.8897*\n*",  
 "Epoch 54/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7303 - acc: 0.9131*\n*",  
 "Epoch 55/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.6972 - acc: 0.8991*\n*",  
 "Epoch 56/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7531 - acc: 0.9225*\n*",  
 "Epoch 57/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6939 - acc: 0.9061*\n*",  
 "Epoch 58/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7237 - acc: 0.9108*\n*",  
 "Epoch 59/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6571 - acc: 0.9131*\n*",  
 "Epoch 60/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.8023 - acc: 0.9038*\n*",  
 "Epoch 61/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.6355 - acc: 0.9178*\n*",  
 "Epoch 62/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6570 - acc: 0.8920*\n*",  
 "Epoch 63/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6835 - acc: 0.9085*\n*",  
 "Epoch 64/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.6536 - acc: 0.9249*\n*",  
 "Epoch 65/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6585 - acc: 0.9178*\n*",  
 "Epoch 66/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6386 - acc: 0.9155*\n*",  
 "Epoch 67/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.7730 - acc: 0.9038*\n*",  
 "Epoch 68/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.6834 - acc: 0.8991*\n*",  
 "Epoch 69/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6471 - acc: 0.9108*\n*",  
 "Epoch 70/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5885 - acc: 0.9249*\n*",  
 "Epoch 71/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5928 - acc: 0.9155*\n*",  
 "Epoch 72/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.7023 - acc: 0.9131*\n*",  
 "Epoch 73/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5818 - acc: 0.9178*\n*",  
 "Epoch 74/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5630 - acc: 0.9155*\n*",  
 "Epoch 75/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.5795 - acc: 0.9108*\n*",  
 "Epoch 76/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5824 - acc: 0.9061*\n*",  
 "Epoch 77/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6017 - acc: 0.9155*\n*",  
 "Epoch 78/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.5646 - acc: 0.9202*\n*",  
 "Epoch 79/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5415 - acc: 0.9155*\n*",  
 "Epoch 80/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5451 - acc: 0.9249*\n*",  
 "Epoch 81/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5475 - acc: 0.9061*\n*",  
 "Epoch 82/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5329 - acc: 0.9319*\n*",  
 "Epoch 83/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6155 - acc: 0.9061*\n*",  
 "Epoch 84/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5344 - acc: 0.9131*\n*",  
 "Epoch 85/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4816 - acc: 0.9249*\n*",  
 "Epoch 86/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5469 - acc: 0.9202*\n*",  
 "Epoch 87/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4869 - acc: 0.9225*\n*",  
 "Epoch 88/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.4661 - acc: 0.9272*\n*",  
 "Epoch 89/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.5022 - acc: 0.9108*\n*",  
 "Epoch 90/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.4861 - acc: 0.9061*\n*",  
 "Epoch 91/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6074 - acc: 0.9178*\n*",  
 "Epoch 92/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.6664 - acc: 0.9061*\n*",  
 "Epoch 93/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.4741 - acc: 0.9296*\n*",  
 "Epoch 94/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4954 - acc: 0.9155*\n*",  
 "Epoch 95/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4736 - acc: 0.9225*\n*",  
 "Epoch 96/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.4443 - acc: 0.9343*\n*",  
 "Epoch 97/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.4802 - acc: 0.9202*\n*",  
 "Epoch 98/100*\n*",  
 "14/14 [==============================] - 0s 2ms/step - loss: 0.4229 - acc: 0.9225*\n*",  
 "Epoch 99/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.5408 - acc: 0.9131*\n*",  
 "Epoch 100/100*\n*",  
 "14/14 [==============================] - 0s 3ms/step - loss: 0.3975 - acc: 0.9272*\n*",  
 "Model: *\"*sequential\_45*\"\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 " Layer (type) Output Shape Param # *\n*",  
 "=================================================================*\n*",  
 " dense\_104 (Dense) (None, 20) 620 *\n*",  
 " *\n*",  
 " dense\_105 (Dense) (None, 1) 21 *\n*",  
 " *\n*",  
 "=================================================================*\n*",  
 "Total params: 641*\n*",  
 "Trainable params: 641*\n*",  
 "Non-trainable params: 0*\n*",  
 "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\n*",  
 "None*\n*",  
 "5/5 [==============================] - 0s 3ms/step - loss: 1.3143 - acc: 0.7902*\n*",  
 "[1.314283013343811, 0.7902097702026367]*\n*"  
 ]  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "from keras.datasets import mnist*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers import Dense, Dropout*\n*",  
 "import matplotlib.pyplot as plt*\n*",  
 "*\n*",  
 "# load MNIST dataset*\n*",  
 "(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()*\n*",  
 "*\n*",  
 "# normalize pixel values to range [0, 1]*\n*",  
 "x\_train = x\_train.astype('float32') / 255*\n*",  
 "x\_test = x\_test.astype('float32') / 255*\n*",  
 "*\n*",  
 "# convert class labels to binary class matrices*\n*",  
 "num\_classes = 10*\n*",  
 "y\_train = keras.utils.to\_categorical(y\_train, num\_classes)*\n*",  
 "y\_test = keras.utils.to\_categorical(y\_test, num\_classes)*\n*",  
 "*\n*",  
 "# create a simple neural network model*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='relu', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(512, activation='relu'))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "*\n*",  
 "model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])*\n*",  
 "*\n*",  
 "# train the model and record the training history*\n*",  
 "history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),*\n*",  
 " epochs=20, batch\_size=128)*\n*",  
 "*\n*",  
 "# plot the training and validation accuracy and loss curves*\n*",  
 "plt.figure(figsize=(10, 5))*\n*",  
 "plt.subplot(1, 2, 1)*\n*",  
 "plt.plot(history.history['accuracy'])*\n*",  
 "plt.plot(history.history['val\_accuracy'])*\n*",  
 "plt.title('Model Accuracy')*\n*",  
 "plt.ylabel('Accuracy')*\n*",  
 "plt.xlabel('Epoch')*\n*",  
 "plt.legend(['Train', 'Validation'], loc='lower right')*\n*",  
 "*\n*",  
 "plt.subplot(1, 2, 2)*\n*",  
 "plt.plot(history.history['loss'])*\n*",  
 "plt.plot(history.history['val\_loss'])*\n*",  
 "plt.title('Model Loss')*\n*",  
 "plt.ylabel('Loss')*\n*",  
 "plt.xlabel('Epoch')*\n*",  
 "plt.legend(['Train', 'Validation'], loc='upper right')*\n*",  
 "*\n*",  
 "plt.show()*\n*"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/",  
 "height": *1000* },  
 "id": "NOyFYu16tb2X",  
 "outputId": "c293e69c-64f9-4c71-95d2-26cecc64c8d6"  
 },  
 "execution\_count": *84*,  
 "outputs": [  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "Epoch 1/20*\n*",  
 "469/469 [==============================] - 16s 27ms/step - loss: 0.2524 - accuracy: 0.9232 - val\_loss: 0.1042 - val\_accuracy: 0.9650*\n*",  
 "Epoch 2/20*\n*",  
 "469/469 [==============================] - 17s 36ms/step - loss: 0.1024 - accuracy: 0.9684 - val\_loss: 0.0823 - val\_accuracy: 0.9742*\n*",  
 "Epoch 3/20*\n*",  
 "469/469 [==============================] - 14s 29ms/step - loss: 0.0713 - accuracy: 0.9773 - val\_loss: 0.0733 - val\_accuracy: 0.9778*\n*",  
 "Epoch 4/20*\n*",  
 "469/469 [==============================] - 13s 28ms/step - loss: 0.0554 - accuracy: 0.9823 - val\_loss: 0.0632 - val\_accuracy: 0.9801*\n*",  
 "Epoch 5/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0468 - accuracy: 0.9847 - val\_loss: 0.0651 - val\_accuracy: 0.9812*\n*",  
 "Epoch 6/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0379 - accuracy: 0.9875 - val\_loss: 0.0620 - val\_accuracy: 0.9821*\n*",  
 "Epoch 7/20*\n*",  
 "469/469 [==============================] - 13s 28ms/step - loss: 0.0330 - accuracy: 0.9894 - val\_loss: 0.0815 - val\_accuracy: 0.9755*\n*",  
 "Epoch 8/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0325 - accuracy: 0.9894 - val\_loss: 0.0749 - val\_accuracy: 0.9796*\n*",  
 "Epoch 9/20*\n*",  
 "469/469 [==============================] - 15s 31ms/step - loss: 0.0269 - accuracy: 0.9909 - val\_loss: 0.0743 - val\_accuracy: 0.9806*\n*",  
 "Epoch 10/20*\n*",  
 "469/469 [==============================] - 12s 27ms/step - loss: 0.0247 - accuracy: 0.9916 - val\_loss: 0.0694 - val\_accuracy: 0.9825*\n*",  
 "Epoch 11/20*\n*",  
 "469/469 [==============================] - 14s 31ms/step - loss: 0.0246 - accuracy: 0.9920 - val\_loss: 0.0723 - val\_accuracy: 0.9814*\n*",  
 "Epoch 12/20*\n*",  
 "469/469 [==============================] - 12s 26ms/step - loss: 0.0217 - accuracy: 0.9922 - val\_loss: 0.0688 - val\_accuracy: 0.9827*\n*",  
 "Epoch 13/20*\n*",  
 "469/469 [==============================] - 16s 35ms/step - loss: 0.0201 - accuracy: 0.9932 - val\_loss: 0.0741 - val\_accuracy: 0.9810*\n*",  
 "Epoch 14/20*\n*",  
 "469/469 [==============================] - 12s 26ms/step - loss: 0.0203 - accuracy: 0.9930 - val\_loss: 0.0720 - val\_accuracy: 0.9819*\n*",  
 "Epoch 15/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0190 - accuracy: 0.9934 - val\_loss: 0.0660 - val\_accuracy: 0.9829*\n*",  
 "Epoch 16/20*\n*",  
 "469/469 [==============================] - 11s 23ms/step - loss: 0.0176 - accuracy: 0.9942 - val\_loss: 0.0729 - val\_accuracy: 0.9834*\n*",  
 "Epoch 17/20*\n*",  
 "469/469 [==============================] - 11s 23ms/step - loss: 0.0165 - accuracy: 0.9948 - val\_loss: 0.0690 - val\_accuracy: 0.9841*\n*",  
 "Epoch 18/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0164 - accuracy: 0.9945 - val\_loss: 0.0704 - val\_accuracy: 0.9843*\n*",  
 "Epoch 19/20*\n*",  
 "469/469 [==============================] - 12s 26ms/step - loss: 0.0164 - accuracy: 0.9949 - val\_loss: 0.0734 - val\_accuracy: 0.9835*\n*",  
 "Epoch 20/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0131 - accuracy: 0.9958 - val\_loss: 0.0752 - val\_accuracy: 0.9844*\n*"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 720x360 with 2 Axes>"  
 ],  
 "image/png": "\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras\n",  
 "from keras.datasets import mnist\n",  
 "from keras.models import Sequential\n",  
 "from keras.layers import Dense, Dropout\n",  
 "import matplotlib.pyplot as plt\n",  
 "import numpy as np\n",  
 "\n",  
 "# load MNIST dataset\n",  
 "(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()\n",  
 "\n",  
 "# normalize pixel values to range [0, 1]\n",  
 "x\_train = x\_train.astype('float32') / 255\n",  
 "x\_test = x\_test.astype('float32') / 255\n",  
 "\n",  
 "# convert class labels to binary class matrices\n",  
 "num\_classes = 10\n",  
 "y\_train = keras.utils.to\_categorical(y\_train, num\_classes)\n",  
 "y\_test = keras.utils.to\_categorical(y\_test, num\_classes)\n",  
 "\n",  
 "# create a simple neural network model\n",  
 "model = Sequential()\n",  
 "model.add(Dense(512, activation='relu', input\_shape=(784,)))\n",  
 "model.add(Dropout(0.2))\n",  
 "model.add(Dense(512, activation='relu'))\n",  
 "model.add(Dropout(0.2))\n",  
 "model.add(Dense(num\_classes, activation='softmax'))\n",  
 "\n",  
 "model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])\n",  
 "\n",  
 "# train the model\n",  
 "model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),\n",  
 " epochs=20, batch\_size=128)\n",  
 "\n",  
 "# plot one of the images in the test data\n",  
 "plt.imshow(x\_test[0], cmap='gray')\n",  
 "plt.show()\n",  
 "\n",  
 "# make a prediction on the image using the trained model\n",  
 "prediction = model.predict(x\_test[0].reshape(1, -1))\n",  
 "print('Model prediction:', np.argmax(prediction))\n"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/",  
 "height": *999* },  
 "id": "VmYFqXgJuL-B",  
 "outputId": "4e131bb3-4a8b-4b6b-a622-22459dc31300"  
 },  
 "execution\_count": *85*,  
 "outputs": [  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "Epoch 1/20*\n*",  
 "469/469 [==============================] - 12s 22ms/step - loss: 0.2488 - accuracy: 0.9253 - val\_loss: 0.1118 - val\_accuracy: 0.9652*\n*",  
 "Epoch 2/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.1016 - accuracy: 0.9684 - val\_loss: 0.0742 - val\_accuracy: 0.9769*\n*",  
 "Epoch 3/20*\n*",  
 "469/469 [==============================] - 15s 31ms/step - loss: 0.0713 - accuracy: 0.9779 - val\_loss: 0.0695 - val\_accuracy: 0.9784*\n*",  
 "Epoch 4/20*\n*",  
 "469/469 [==============================] - 14s 30ms/step - loss: 0.0561 - accuracy: 0.9819 - val\_loss: 0.0702 - val\_accuracy: 0.9779*\n*",  
 "Epoch 5/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0455 - accuracy: 0.9855 - val\_loss: 0.0615 - val\_accuracy: 0.9822*\n*",  
 "Epoch 6/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0381 - accuracy: 0.9873 - val\_loss: 0.0648 - val\_accuracy: 0.9818*\n*",  
 "Epoch 7/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0337 - accuracy: 0.9891 - val\_loss: 0.0732 - val\_accuracy: 0.9810*\n*",  
 "Epoch 8/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0296 - accuracy: 0.9905 - val\_loss: 0.0675 - val\_accuracy: 0.9811*\n*",  
 "Epoch 9/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0279 - accuracy: 0.9905 - val\_loss: 0.0756 - val\_accuracy: 0.9799*\n*",  
 "Epoch 10/20*\n*",  
 "469/469 [==============================] - 12s 26ms/step - loss: 0.0248 - accuracy: 0.9915 - val\_loss: 0.0804 - val\_accuracy: 0.9806*\n*",  
 "Epoch 11/20*\n*",  
 "469/469 [==============================] - 11s 23ms/step - loss: 0.0238 - accuracy: 0.9918 - val\_loss: 0.0753 - val\_accuracy: 0.9807*\n*",  
 "Epoch 12/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0237 - accuracy: 0.9923 - val\_loss: 0.0743 - val\_accuracy: 0.9814*\n*",  
 "Epoch 13/20*\n*",  
 "469/469 [==============================] - 12s 25ms/step - loss: 0.0211 - accuracy: 0.9929 - val\_loss: 0.0785 - val\_accuracy: 0.9803*\n*",  
 "Epoch 14/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0168 - accuracy: 0.9945 - val\_loss: 0.0694 - val\_accuracy: 0.9838*\n*",  
 "Epoch 15/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0192 - accuracy: 0.9934 - val\_loss: 0.0786 - val\_accuracy: 0.9820*\n*",  
 "Epoch 16/20*\n*",  
 "469/469 [==============================] - 11s 23ms/step - loss: 0.0184 - accuracy: 0.9938 - val\_loss: 0.0768 - val\_accuracy: 0.9827*\n*",  
 "Epoch 17/20*\n*",  
 "469/469 [==============================] - 11s 23ms/step - loss: 0.0164 - accuracy: 0.9948 - val\_loss: 0.0775 - val\_accuracy: 0.9823*\n*",  
 "Epoch 18/20*\n*",  
 "469/469 [==============================] - 10s 22ms/step - loss: 0.0162 - accuracy: 0.9948 - val\_loss: 0.0800 - val\_accuracy: 0.9822*\n*",  
 "Epoch 19/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0145 - accuracy: 0.9951 - val\_loss: 0.0873 - val\_accuracy: 0.9820*\n*",  
 "Epoch 20/20*\n*",  
 "469/469 [==============================] - 11s 24ms/step - loss: 0.0140 - accuracy: 0.9957 - val\_loss: 0.0807 - val\_accuracy: 0.9841*\n*"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 432x288 with 1 Axes>"  
 ],  
 "image/png": "*\n*"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "1/1 [==============================] - 0s 120ms/step*\n*",  
 "Model prediction: 7*\n*"  
 ]  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "from keras.datasets import mnist*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers import Dense, Dropout*\n*",  
 "import matplotlib.pyplot as plt*\n*",  
 "import numpy as np*\n*",  
 "*\n*",  
 "# load MNIST dataset*\n*",  
 "(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()*\n*",  
 "*\n*",  
 "# normalize pixel values to range [0, 1]*\n*",  
 "x\_train = x\_train.astype('float32') / 255*\n*",  
 "x\_test = x\_test.astype('float32') / 255*\n*",  
 "*\n*",  
 "# convert class labels to binary class matrices*\n*",  
 "num\_classes = 10*\n*",  
 "y\_train = keras.utils.to\_categorical(y\_train, num\_classes)*\n*",  
 "y\_test = keras.utils.to\_categorical(y\_test, num\_classes)*\n*",  
 "*\n*",  
 "# create a list of models to train*\n*",  
 "models = []*\n*",  
 "*\n*",  
 "# model with 1 hidden layer and tanh activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='tanh', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('1 hidden layer with tanh', model))*\n*",  
 "*\n*",  
 "# model with 1 hidden layer and sigmoid activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('1 hidden layer with sigmoid', model))*\n*",  
 "*\n*",  
 "# model with 2 hidden layers and tanh activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='tanh', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(512, activation='tanh'))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('2 hidden layers with tanh', model))*\n*",  
 "*\n*",  
 "# model with 2 hidden layers and sigmoid activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(512, activation='sigmoid'))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('2 hidden layers with sigmoid', model))*\n*",  
 "*\n*",  
 "# train each model and plot loss and accuracy curves*\n*",  
 "for name, model in models:*\n*",  
 " model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])*\n*",  
 " history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),*\n*",  
 " epochs=20, batch\_size=128, verbose=0)*\n*",  
 " # plot loss and accuracy curves*\n*",  
 " plt.plot(history.history['loss'], label='train\_loss')*\n*",  
 " plt.plot(history.history['val\_loss'], label='val\_loss')*\n*",  
 " plt.plot(history.history['accuracy'], label='train\_accuracy')*\n*",  
 " plt.plot(history.history['val\_accuracy'], label='val\_accuracy')*\n*",  
 " plt.title(name)*\n*",  
 " plt.xlabel('Epoch')*\n*",  
 " plt.legend()*\n*",  
 " plt.show()*\n*",  
 " *\n*",  
 " # evaluate the model on test data*\n*",  
 " loss, accuracy = model.evaluate(x\_test.reshape(-1, 784), y\_test, verbose=0)*\n*",  
 " print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))*\n*"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/",  
 "height": *1000* },  
 "id": "0\_0j3dDGuxMT",  
 "outputId": "b557ff75-4e93-4fb3-e837-88f6ef517d3b"  
 },  
 "execution\_count": *88*,  
 "outputs": [  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 432x288 with 1 Axes>"  
 ],  
 "image/png": "\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "*1* hidden layer with tanh - Test loss: *0.0716*, Test accuracy: *0.9809*\n"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size *432*x*288* with *1* Axes>"  
 ],  
 "image/png": "iVBORw*0*KGgoAAAANSUhEUgAAAXQAAAEWCAYAAAB*2*X*2*wCAAAAOXRFWHRTb*2*Z*0*d*2*FyZQBNYXRwbG*90*bGliIHZlcnNpb*24*zLjUuMywgaHR*0*cHM*6*Ly*9*tYXRwbG*90*bGliLm*9*yZy/NK*7*nSAAAACXBIWXMAAAsTAAALEwEAmpwYAAA/yElEQVR*4*nO*3*dd*3*gc*1*\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "1 hidden layer with sigmoid - Test loss: 0.0642, Test accuracy: 0.9809*\n*"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 432x288 with 1 Axes>"  
 ],  
 "image/png": "\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "*2* hidden layers with tanh - Test loss: *0.0686*, Test accuracy: *0.9808*\n"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size *432*x*288* with *1* Axes>"  
 ],  
 "image/png": "iVBORw*0*KGgoAAAANSUhEUgAAAXQAAAEWCAYAAAB*2*X*2*wCAAAAOXRFWHRTb*2*Z*0*d*2*FyZQBNYXRwbG*90*bGliIHZlcnNpb*24*zLjUuMywgaHR*0*cHM*6*Ly*9*tYXRwbG*90*bGliLm*9*yZy/NK*7*nSAAAACXBIWXMAAAsTAAALEwEAmpwYAABAx*0*lEQVR*4*nO*3*deZxVdf*348*df*7*rrMyG/sOwyogIAgIKqRlaASlIZr*6*cyltca/*8*ytesrKyvlZVZpmlpSpapZZlZ*5*gKS*4*oYLisi+yIAswyzMMMvd*3*r*8*/zpnhznBn*5*gIzc*2*cu*7*yecxz*3*ncz*7*nnPc*99877*nPu*5536*OqCrGGGO*6*P*0*+qAzDGGNM+LKEbY*0*yasIRujDFpwhK*6*McakCUvoxhiTJiyhG*2*NMmrCEfowQkUtE*5*KVW*5*v*9*LRC*5*uYd*5*QEVER*8*bUw/xYR+UN*7*xRq*33*jkiUtLe*602*F*1*\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "2 hidden layers with sigmoid - Test loss: 0.0663, Test accuracy: 0.9830*\n*"  
 ]  
 }  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "source": [  
 "import keras*\n*",  
 "from keras.datasets import mnist*\n*",  
 "from keras.models import Sequential*\n*",  
 "from keras.layers import Dense, Dropout*\n*",  
 "import matplotlib.pyplot as plt*\n*",  
 "import numpy as np*\n*",  
 "*\n*",  
 "# load MNIST dataset*\n*",  
 "(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()*\n*",  
 "*\n*",  
 "# convert class labels to binary class matrices*\n*",  
 "num\_classes = 10*\n*",  
 "y\_train = keras.utils.to\_categorical(y\_train, num\_classes)*\n*",  
 "y\_test = keras.utils.to\_categorical(y\_test, num\_classes)*\n*",  
 "*\n*",  
 "# create a list of models to train*\n*",  
 "models = []*\n*",  
 "*\n*",  
 "# model with 1 hidden layer and tanh activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='tanh', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('1 hidden layer with tanh', model))*\n*",  
 "*\n*",  
 "# model with 1 hidden layer and sigmoid activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('1 hidden layer with sigmoid', model))*\n*",  
 "*\n*",  
 "# model with 2 hidden layers and tanh activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='tanh', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(512, activation='tanh'))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('2 hidden layers with tanh', model))*\n*",  
 "*\n*",  
 "# model with 2 hidden layers and sigmoid activation*\n*",  
 "model = Sequential()*\n*",  
 "model.add(Dense(512, activation='sigmoid', input\_shape=(784,)))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(512, activation='sigmoid'))*\n*",  
 "model.add(Dropout(0.2))*\n*",  
 "model.add(Dense(num\_classes, activation='softmax'))*\n*",  
 "models.append(('2 hidden layers with sigmoid', model))*\n*",  
 "*\n*",  
 "# train each model and plot loss and accuracy curves*\n*",  
 "for name, model in models:*\n*",  
 " model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])*\n*",  
 " history = model.fit(x\_train.reshape(-1, 784), y\_train, validation\_data=(x\_test.reshape(-1, 784), y\_test),*\n*",  
 " epochs=20, batch\_size=128, verbose=0)*\n*",  
 " # plot loss and accuracy curves*\n*",  
 " plt.plot(history.history['loss'], label='train\_loss')*\n*",  
 " plt.plot(history.history['val\_loss'], label='val\_loss')*\n*",  
 " plt.plot(history.history['accuracy'], label='train\_accuracy')*\n*",  
 " plt.plot(history.history['val\_accuracy'], label='val\_accuracy')*\n*",  
 " plt.title(name)*\n*",  
 " plt.xlabel('Epoch')*\n*",  
 " plt.legend()*\n*",  
 " plt.show()*\n*",  
 " *\n*",  
 " # evaluate the model on test data*\n*",  
 " loss, accuracy = model.evaluate(x\_test.reshape(-1, 784), y\_test, verbose=0)*\n*",  
 " print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))*\n*"  
 ],  
 "metadata": {  
 "colab": {  
 "base\_uri": "https://localhost:8080/",  
 "height": *1000* },  
 "id": "9BaQ4jKJwQVD",  
 "outputId": "97c848f3-387c-4de3-dc18-e538158287df"  
 },  
 "execution\_count": *89*,  
 "outputs": [  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 432x288 with 1 Axes>"  
 ],  
 "image/png": "\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "*1* hidden layer with tanh - Test loss: *0.1895*, Test accuracy: *0.9439*\n"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size *432*x*288* with *1* Axes>"  
 ],  
 "image/png": "iVBORw*0*KGgoAAAANSUhEUgAAAXQAAAEWCAYAAAB*2*X*2*wCAAAAOXRFWHRTb*2*Z*0*d*2*FyZQBNYXRwbG*90*bGliIHZlcnNpb*24*zLjUuMywgaHR*0*cHM*6*Ly*9*tYXRwbG*90*bGliLm*9*yZy/NK*7*nSAAAACXBIWXMAAAsTAAALEwEAmpwYAABA*2*UlEQVR*4*nO*3*dd*5*xU*9*dX*48*c+Zsp*2*tLHVpRrDACggixhqNisaojwaxPpaoj*4*k*9*ib+gMcbHEk*0*ziZHEklgfE*2*tMiCExGlEsWFZFUIKAiOyiwnbYOu*38*/rh*3*ZmeXLbOwu*7*M*7*nPe+*7*uu*279*x*7*ZnbmzHe+*997*vFVXFGGPM*0*OdJdgDGGGP*6*hiV*0*Y*4*xJEZbQjTEmRVhCN*8*aYFGEJ*3*RhjUoQldGOMSRGW*0*HdTInKeiLzazfp/iMi*5*XaybKCIqIr*4*u*1*t*8*oIv/XV*7*HGbfcIEano*6*+*32*FxEZLyINIuLtpoyKyJ*79*vZ/+*0*NN+++t*9*YLpmCT*2*FiMhlIlImIq*0*i*8*uCubEtVj*1*PVh/ootN*2*Sqm*5*S*1*RxVDQOIyEsicmF/*72*egJGu/pmud*1*rDMkPUZcAtwLJCZ*5*FhSgoj*4*VDWU*7*DiMSYTV*0*FOIqv*5*ZVf*8*CVCf*6*GBH*5*uYjUisgnInJc*3*PJYbVJEvG*65*KhHZAHytwzYmicjLIrJdRJ*4*HhndYP*1*dEXheROhF*5*X*0*SO*6*LCfm*0*XkNffx/xKRdo/vJvaFIvKx+*7*jVIvJf*7*vI*0*EakRkdK*4*siNEpElEit*35E0*RkhRvT*6*yKyX*1*zZjSLyfRFZCTR*2*bFoSkf*8*Vkd+*4034*RaRSRn*7*nzmSLSIiKF*8*U*1*TInIrcChwl*9*tMcVfcJr*8*qIuvcWBaJiHTxfOe*4*v*8*C*2*icgWEbnDXd*6*uCcz*9*fyxzX*5*cX*3*G*3*+X*4*ey*54*tIufu/v*0*REDhCRlW*4*Md*8*Xt*0*yMi*14*vIpyKyVUQeFpG*8*bvbb*5*fvADABVtSHFBpxa+oM*9*lDkPCAIXAV*7*gWzg*1*fHHXvwRc*6E5*fAqwBxgGFwFJAAZ+*7*fjlwB*5*AOHAZsB/*7*PXTcW*5*wvmeJwKxNHufHHcfj*4*GpuD*8*qngJuL*2*LmI*8*AKuLm*5*wNj*3*O*0*uABqB*0e663*wI/iSt*7*JfA*3*d*3*omsBU*40*H*3*u*5*wIbgXR*3*/UZghft*8*MzuJ*40*hglTv*9*ZTf+N+PWve*9*OT+zwOsVe*07*htKfAskA+MByqBeV*08*/+XAOe*50*DjC*3*i/*0*sB*34*OpAGHANvi/h/RsncDGcAxQAvwF*2*\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "1 hidden layer with sigmoid - Test loss: 0.1420, Test accuracy: 0.9582*\n*"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size 432x288 with 1 Axes>"  
 ],  
 "image/png": "\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "*2* hidden layers with tanh - Test loss: *0.1422*, Test accuracy: *0.9563*\n"  
 ]  
 },  
 {  
 "output\_type": "display\_data",  
 "data": {  
 "text/plain": [  
 "<Figure size *432*x*288* with *1* Axes>"  
 ],  
 "image/png": "iVBORw*0*KGgoAAAANSUhEUgAAAXQAAAEWCAYAAAB*2*X*2*wCAAAAOXRFWHRTb*2*Z*0*d*2*FyZQBNYXRwbG*90*bGliIHZlcnNpb*24*zLjUuMywgaHR*0*cHM*6*Ly*9*tYXRwbG*90*bGliLm*9*yZy/NK*7*nSAAAACXBIWXMAAAsTAAALEwEAmpwYAABA+klEQVR*4*nO*3*deXxV*9*Z*3*/*8*dfnLrk*3*NzsJCZCwqiBCBASVimsdW*2*uttlZkrDpqWx*2*tdel*0*/JVxujitnWlnujpjtdpRq*2*Xq*1*to*61*o*6*tCtK*6*AyKbyo*4*JCCSQfbnr*5*/fHOUluQjYgyU*1*uPk*8e53*G*27*z*3*ne*08*u*7*/u*933*PuuaKqGGOMGfk*8*qa*6*AMcaYgWGBbowxacIC*3*Rhj*0*oQFujHGpAkLdGOMSRMW*6*MYYkyYs*0*EcJEblGRP*7*ay/o/isjVPaybIiIqIr*4e1*t*8*pIr*8*aqLombfdsEakc*6*O*2*mQm/H*113*/sIjcNdj*7*GSxH*8*/oxA*8*cO*8*AglIgHgZ*8*DfAGOAbcA/qeofj*2*R*7*qvqJAaye*6*SL*5*+IrINcAXVfX*0*wdzPULLXz/BgLfSRywdUAGcBecDXgSdEZEoqKzXSWSvSjGQW*6*COUqjap*6*p*2*qulNVE*6*r*6*LLADmN/b*40*TkByJSIyI*7*RCS*51*bhCRL*7*oTnvdctUish*34*ZJdtTBWRl*0*WkQUT+DBR*1*Wb*9*QRF*4*VkVoReUdEzu*6*yn++IyCvu*4*/*8*kIp*0e30*vdl*4*rINvdxm*0*TkM+*7*yDBE*5*KCLlSWWLRaRZRMa*68*xeKyFq*3*Tq+KyIlJZXeKyNdEZB*3*QJCI+d*363*u*6*/*3*ReTcbuoz*1*d*2*ex*51*/QET*2*J*61*/VERuSz*6*+IjITuA/*4*iIg*0*ikht*0*iYLROQP*7*j*7*fEJFjejgOQRH*5*lYgccPf/loiUJO/HnfaKyA/dv+MOEflycteHW/Yu*93*g*0*isj/ikihiCwTkXp*3*u*1*OS*9*nuau*6*zOHZ/W*5*e/ar*9*ePGUSqakMaDEAJ*0*Aoc*38*P*6*a*4*AocB*3*gBW*4E9*gDirl+B*0*w*0*AcAPwHjARpztnOaCAz*13*/GvAjIACcCTQAv*3*LXlQIHgAtwGgznufNjk/azDZgOZLrz*3*+uhzmcDlUnzi*4*EJ*7*naXAE*3*AeHfdz*4*DvJ*5*W*9*Ffhfd*3*oesB*841*X*3*uVwM*7*gYC*7*fiew*1*n*2*+mcAMnE*8*/E*9*z*1*U*4*BjeqjjB*8*B*8*d/p*9*YDswM*2*ndvG*6*O*7*zXAX*7*ts*52*H*3*OJ*2*C*8*+lrGfBYD/v*8*e+B/gZD*7*fOYDuT*38*HTcBZUAB*8*EKXv+MKYCtwDM*6*nvE*3*AZpxuPB/wCPCQW*3*YMUANc*5*a*673*J*0*vPNzXjw*2*DN*1*gLPQ*2*IiB*8*nAH*6*pqu/*1*UnSXqj*6*gqnHgl*8*B*4*nDeCri*4*DfqKqFap*6*EPi*3*pH*1*NAk*4*GvqGqYVVdiRMuba*4*EnlPV*59*T*55*PBnYBVOwLd*5*SFU*3*q*2*oL*8*AQwtz/PU*1*WfVNU*97*nYfB*7*bgBCDu*87*lcRMSdvwp*41*J*2*+Hvi*5*qr*6*hqnFV/SUQBhYmbf*5*u*9*/m*2*AHGcN*6*sTRMSvzqegbT*1*U*62*XgLBEZ*584*/*5*c*5*PBXKBd/rz*3*FxPq+qbqhrD+XvO*7*aFcFCgEjnWfz*2*pVre+m*3*GXAT*1*W*1*UlVrgO*91*U+YhVd*2*mqnXAH*4*FtqvqCW*4*cncd*4*MwWllb*1*HVR*1*U*1*pqq/xgntT/Ww*325*fP*2*ZwWaCPcO*7*H/UeBCPDlPorvbZtQ*1*WZ*3*MrubchNwWqhtdnVZV*6*OqTT*2*snwwsdrsCat*0*uhdNx*3*jwOqQfQ*3*EMdDiEif*5*fUbVILzMbt*7*lHVN*9*xtnS*0*ixwPHAs*8*k*1*emrXeo*00*X*0*ubdqfr*6*puBW*4*D*7*gT*2*i*8*hjIpJcNtnLOJ*8*kzgRW*4*rRUz*3*KHv*6*hqoj/PzdXf*4*/Io*8*DzwmIjsEZF/d*9*/Uu+r*6*d*6*zopsy+pOmWbubb*6*jCBzn*9*n*3*PnSfuy*36*+PMILFAH*8*Hc*1*uh/*47*SyP*6*uq*0*QHa*9*Ic*4*gddmUpd*1*BSKS*1*cP*6*CuBRVc*1*PGrJUtbvWYb+JyGTgAZw*3*rUJVzQc*2*AJJU*7*Jc*4*nxCuAp*5*S*1*dakOn*23*S*51*CbiuzTafbjqrq/*6*hzFcpkd*933e6*jay*8*AZOKH+MvBXYBFOoL/cw*2*OO*6*hanqhpV*1*X*9*R*1*ROA*04*ALgb/rpuiHON*0*tbSZ*2*U*6*a/*9*uAci*2*STgN*097*Len*148*ZRBboI*9*u*9*wEzgU*25*XwUB*5*ArhFRMpEpABY*2*rZCVXfhdKH*8*i*3*sy*8*nQ*6*f+z+FfApEfm*4e3*IsKM*715*GUcnSycIKwCEJFrcVroyX*4*FfAYn*1*B*9*JWv*4*AcIOInCqOLBH*5*pIjkdLcjEZkhIh*8*V*59*LQVpyWarctbVXd*4*q*6*/EnjZ*7*frYB*3*yWngN*9*H*1*AmIhl*9*Peke*6*neOiJSLiBeox+mC*6*a*5*+TwC*3*ikipiOQDXzuS/bmeA*6*aLyOfck*8*ZLgBOAZ*3*vYb*7*evHzO*4*LNBHKLfF+vc*4*/ax*73*asUGkXkigHY/AM*4*H+nfAdYAv+*2*y/nM*4*JxgPAt*8*iKTxVtQK*4*GLgDJ*3*wrgNs*5*yteaqm*4*CfohzQnYfUA*680*qVMhVtfBf*6*StHwVzsng/*8*I*5*kbcV*58*RkTwI*4*/c*3*VON*0*gxcA/*9*VL+ZeCAu/+*2*eXHr*0*p*2*XgI*04*f*7*fqXrbbk*3E4*ffX*1*wLvu/h*7*tptwDwJ+AdcDbOKEcwzlHcFhU*9*QDOJ*4*Gv*4*py*8*/X/AharaXf*37*ev*2*YQdJ*2*hYMxaUFEHgT*2*qOrXU*12*X*4*Uacy*1*TvU*9*WuXScmTdiXKEzacK+ZvoSOKzNGNRHJBM*7*BaaWX*4*HyaejqllTKDyrpcTFoQke/gnCT*9*D*1*Xdker*6*DBMC/AtON*9*PbON*0*z*30*xpjcygsi*4*XY*4*xJE*3220*EXkQRHZLyIbelgvInK*3*iGwVkXUictLAV*9*MYY*0*xf+tOH/jDO*1*QGP*9*LD+E*8*Bx*7*nAqzqV*0*p/a*10*aKiIp*0*yZUq/KmmMMcaxevXqalUd*2926*PgNdVVdK*73*fwuxh*4*RJ*2*+m*9*dFJF*9*Exqvqh*71*td*8*qUKaxataqv*3*RtjjEkiIj*1*+*83*YgToqW*0*vlrvpV*0*/*3*VgROR*6*EVklIquqqqoGYNfGGGPaDOlVLqp*6*v*6*ouUNUFY*8*d*2*+*4*nBGGPMERqIQN*9*N*5*/s*2*lNH*9*/R*2*MMcYMooEI*9*GeAv*3*OvdlkI*1*PXVf*26*MMWbg*9*XlSVER+jXMnuSJxfrD*3*W*4*AfQFXvw*7*k/xAU*498*doBq*4*drMoaY*4*zpWX+ucrm*8*j/UK*3*DRgNTLGGHNE*7*Kv/xhiTJuzmXMaYUU*1*VIRolEQ*6*j*4*TCJ*1*jAaCaOtre*3*LOi*0*Ph*9*F*4*AjSBJhKQUEgkAO*2*YT*16*nCdDO*67*LPOYfM*8*vI+*63*a*4*LNCNGUQaiRCvqyNWU*0*O*8*ppZ*4*bS*3*x+jrE*40*F*8*PvD*6*EJ*8*P*8*TtjfD*7E22*Xe*53*fmvd*72*ZRqJOCETDqPhSHvQJM*8*nWls*7*ptuWh*8*NoNIpkZOAJBpBgZsc*4*M*4*gEgp*3*HwSAed*5*DMTDyBABIIOPtqbibR*0*uIMzc*1*o+*3*QLiZbmjmXN*7*vKWtjKtaCwG*8*RgaT*6*DxGMTiaDwOcWes*8*RjEE*86*yWAxNJJVLOL/lIQAizuDxuNMgSMdyd*5*DkeUCjUSewIxG*0*tRWG*8*p*5*WIviKiy*3*QzeigqmhzM/HGJhJNbUNjx*3*Sz++NMIuBp+*8*/q*6*Tzv*8*QA*9*zHs*87*QHQPu*3*xONPiccp*4*vSAexOM+*1*uPtmFYlXldPvLbGCegaZ*9*wptN*1*licbGVB*7*KDiJOOGdkQIYfjUTRlhY*0*Ehn*8*fft*8*eEIhPJmZeDIzkcxMxO*93*jrfP*67*yBZWQ*4*b*25*eL*3*i*97*puX*11*mWVA*6*vB/F*4*ne*2*qAs*6*v*3*aPqtn*6*dedU*48*UScRCJpHI+R*0*DgJTZDweYn*7*vSQynHE*8*w*0*vc*5*yGW*4*SXmE*2*J+D*1*GfEPMLUZ*8*Q*8*UPECxGfEvUKHp*8*Pr*9*eP*1*+uMfV*4*fXo*8*Pny*8*Dn*9*fvLnMGvzcDn*69*j*3*uf*1E88*dnFvSW*6*CbAafRKPH*6*euJ*1*dcRr*64*jX*1*XaM*6*+pI*1*NWTaGok*3*hbQycHd*2*EiiuXloW*0*wDwJOVhbegAG*9*+Pt*6*CAjKmTsGbn*48*vaZk*3*vwBvQT*7*enBwneOJxNBZDozE*0*FnVaovF*45*/m*29*fFYx*3*ws*7*rSwAxlIMIhkBJxpt/XsrHOnA*0*HifqFZIzTFmmiMNNISayGWiKEo*8*XiURGsYwmESLa*3*QGiYRbkVa*3*RZ*9*ayvS*6*rTsJRwBd*4*j*7*vcQCPmIBd*5*zhJRrwEM*3*wEs*3*wEsmQjrFXSWiCWCJG*3*A*3*UeCKO*4*iyPaxxVZ*7*p*9*IEFCYyS*0*BVXtKIPz*2*GgiSiQeIRwPE*01*ECcfDROKR*9*mV*6*dD/b*2*iHuDmHwiAef+PB*6*vKgq*0*USUuB*72*jz*8*B*8*I*2*F*3*+CyGZcNTB*2*TWKCPAJpIED*94*kOjefcT*27*SVeU+O*0*cNz/vJ*3*+Q*3*edb*1*+W*4*bRUu*247*Fuuzz*7*D*9*o*3*vYXd/S*4*rRQ*6*+rcwQ*3*qWmc+*0*dTU*85*PxePDk*5*ODNzsaTleUEYX*4*+/tJSPFkhZz*5*pnTMkz*4*fwhEKAtPdNkki*4*+e/*2*Zbb*1*Vyqd+y/bysSdcGjv*64*zHO*7*aTSDjbSCTQ*5*PXJ/aKqePPy*3*aDOx*5*ef*77*Qwe/sbqtISa*6*Eh*2*kRTtIm*4*xjsFXCwRaw+*3*WCJOQiGuQiwhJNRLTJVEQoirh*1*giRnOsicbIPprc*7*TW*2*NNJU*39*QxH*21*snw*7*Hw*0*f+*4*gPn*0*olMd+hJAueXVds+PCF*4*PV*584*sMjns*7*T*4*sXjccaC*4*BFPt*4*MgTlnxINJRzuvxkunLJMObQYY*3*g*4*A*3*gN/jJ+ANtC/L*8*LjLvc*7*y*9*mmPU*8*bv*8*ePz+NqH*5*Pm*26*U*5*l*3*CA/*5*GlrxxtMTGNE*41*FiiRgxjTnjRMxZ*12*V*6*at*7*Uo/ub*9*MACPcU*0*GiVWVdUe*1*tF*9*+*4*jt*3*Ud*0315*i+/YT*27*uXaFUVRKNHvS/JcEPe*60*UjERLhMMSPrIWBz*4*c*3*L*6998*BeX*4*D*1*uOt*78*PDzty/OdcX*4*+*3*nxnmSc*72*/kIPcDaWnHJ/*3*mSh*2*gi*2*v*6*fLp*5*wQhRA*3*Y/szhgUAbyoetC*2*f+pzWnzqlI*9*rnObYQZoiFTRWNdK*0*pyNAGyONNMWaaIokLXPHbfscSAFvgCx/Fln+LLL*92*YT*8*IYpDxUz*1*T*21*f*1*rY+y*59*FdkY*2*IV*8*In*8*fXEZqejmD*1*ihcRaR*978*ODxeJxxUth*6*xYvX*420*fJwd*3*WwiPBh*7*x*4*PF*68*Hv*9*qa*4*KYIE+*4*BKRiNOPWuf*2*pdbVOX*2*p*7*jheV*0*fsYA*2*xfU*5*ox*6*sPHNK*9*IMEg/pISfOPGETp*5*Ab*7*iEnzjSvCPG+dMjylwPpqHwyTCEecEWfJJsUjUaWVHup*8*nFkMynNa*7*JxhwpoOBpI/pbdNBJJDhnBALJK*0*PZuLJCnXb*4*u+PaCJKc*7*SZxmgjjZHGTmHYNQQbIx*3*zbeubo*80*dwa*0*drZ*5*UCvlCTnhmdIRoUWYRIX+ofT*47*I*5*tsfzaZvkz*8*Xr*8*ThuJtD*9*eu*051*CMyk*4*vR*4*vIV+ILH/WsAkSMzxYoPch*0*dRErKqKWHW*1*M*66*qIlZ*9*oD*2*ou*461*paXnjfn*9*ePPz*8*OUX*4*CspIXD*8*DPwl*45*ywLinBVzIO/*7*gSPLm*5*RxyWQ*0*lVaYw*2*UhuupS*5*cR*224*ttN*0*T+OmaC*9*dMi*6*PeDq*1*MLP*92*eQH*8*ynLKSPkD/X*4*sbjrx+iuH*63*bArGtdepcEeF*0*EYhI+*7*K*2499*pedI*4*y+cEdJY/i*5*Av*1*O*3*HcWOG*2*qgMdE*0*knJby/qqkkE*6*arqoiXuUEeKK*5*+dANeL*1*ON*4*LbneAfP*57*gzJmduxfa*1*ud*1*TEvoyFu*1*gyUcD*9*MQaaA+Uk*9*DpIHGSGPn+WjHfPK*62*nAt*9*eF*6*Ytpzyzg*3*I*5*f*8*QD*75*gXwKg*4*Uck*3*cMeYE*8*cgO*55*GbkHhLYbS*3*YLH*8*Wmb*7*MYXesjBnuRlygN*7*z*0*EnW/f*8*Y*56*x*9*tO+sfO*3*Q+lnxVQOf*5*tm*6*HrjxZWfjGjsVXVERw*1*gn*4*xo*7*FW*1*TkLBs*7*Fl/RWHzFY*51*wHoR+*4*KOR*0*AQNkQZqw*7*XUtNZ*0*HodrnOlWZ*7*ouXNce*2*NFE*733*zPvGRk*5*FDTkYO*2*RnZ*5*GTkcEzICea*2*sO*5*uOjcj*11*qtxgyxERfo*8*YMHCW/d*2*vElC*7*/*7*xQufDwkEnetV*2*+Z*9*bV/O*6*LIsEMBXVISveGxSWBe*5*V*08*MrWg*8*SnOsmZZYS/vQHO*0837481*kxLtIWmWBN*14*br*2*wD*7*YepC*6*cF*2*Pl*1*D*5*PX*4*KggUUBArID+ZTEiohN*5*BLTkYOuRm*55*Pg*7*wjo*3*I*7*cjwN*3*+XmspGzMyiKboet*8*FCxZoOv*4*EnapSH*6*lnf/P+bod*9*zfs*42*HqwPZx*767*LoTqYvk*0*xfZns*4*dxoH*8*ikIOuMxwTHtyy*2*UjUkfIrJaVRd*0*t*27*EtdBTSVU*52*HqQioYK*9*jbvZX/TfqpaqtjXvK*89*sKuaq*2*iNtx*7*y*2*PxAPsWhYopDxcwYM*4*OQL*0*TIH*2*oP*6*OShu+UhX*4*igLzhqLgczxhw+C/RuJDTB*3*qa*9*bK/bzvba*7*c*7*YHerCdZ*3*KZngy*2*oN*6*duFsiicWMzY*0*lpJQSfvysaGxBLyBFD*0*bY*8*xoMaoDPZqIUlFf*0*Smwt*9*duZ*2*f*9*TlpiHZcfFgQKmJo*3*lfMmn*8e0*vGlMzp*3*MuKxxTl*90*xsi*4*xNAYk/*5*GVaCrKisrV/K*7*rb*9*jW*902*KuorOvVhj*8*sax*7*S*8*acwvmc/UvKkck*38*M*0*/KmURAsSGGtjTGmf*0*ZNoG*88*sJEfrvohb+*19*i*5*JQCbMKZ*3*HupHOZljeNaXnTmJo*3*lZB/*6*K*9*yMcaYgZL*2*gb*67*cTd*3*r*7*mb*53*Y*8*R*0*GggDtOvYNLp*1*+K*32*NfmTbGpJe*0*DfS*6*cB*2*/WP*8*Llr*27*DI*94*uK*78*Oj*4*/+/NkZ*2*SnumrGGDMo*0*i*7*QI/EIj*733*GPevv*5*/*6*cD*2*fOuZT*3*DzvZsZljUt*11*YwxZlClTaCrKs/vep*6*frP*4*Juxt*385*HxH+GrC*77*KjDEzUl*01*Y*4*wZEmkR*6*Gv*2*reGHq*37*Iuup*1*HFdwHPf*9*zX*0*sKl*2*U*6*moZY*8*yQGtGBvqNuBz*9*Z/RNeqniJ*4*sxivn*3*at*7*nomIvsplDGmFFpRAb*6*gZYD*3*PvOvTy*1*+SkC*3*gA*3*z*7*uZq*064*ikxfb*7*+RZYwx*6*W*3*EBfrvt/*6*ef*3*vz*32*iNtXLp*9*Eu*5*Yc*4*NFGUWpbpaxhiTciMu*0*EuySjhl*3*CncNv*82*puVNS*3*V*1*jDFm*2*Bhxgb*5*w/EIWjl+Y*6*moYY*8*ywY/diNcaYNGGBbowxacIC*3*Rhj*0*oQFujHGpAkLdGOMSRMW*6*MYYkyYs*0*I*0*xJk*1*YoBtjTJqwQDfGmDTRr*0*AXkfNF*5*H*0*R*2*SoiS*7*tZP*0*lElovI*2*yKyTkQuGPiqGmOM*6*U*2*fgS*4*iXuAe*4*BPACcDlInJCl*2*JfB*55*Q*1*XnA*3*wI/G+iKGmOM*6*V*1*/WuinAFtVdbuqRoDHgIu*7*lFEg*153*OA/YMXBWNMcb*0*R*38*CvRSoSJqvdJcluxO*4*UkQqgeeAm*7*vbkIhcLyKrRGRVVVXVEVTXGGNMTwbqpOjlwMOqWgZcADwqIodsW*1*XvV*9*UFqrpg*7*NixA*7*RrY*4*wx*0*L*9*A*3*w*1*MTJovc*5*cl+wLwBICqvgYEAfvVCWOMGUL*9*CfS*3*gONEZKqIZOCc*9*HymS*5*kPgHMBRGQmTqBbn*4*oxxgyhPgNdVWPAl*4*HngXdxrmbZKCLfFpGL*3*GJfBa*4*TkXeAXwPXqKoOVqWNMcYcql+/WKSqz+Gc*7*Exe*9*s*2*k*6*U*3*AooGtmjHGmMNh*3*xQ*1*xpg*0*YYFujDFpwgLdGGPShAW*6*McakCQt*0*Y*4*xJExboxhiTJizQjTEmTVigG*2*NMmrBAN*8*aYNGGBbowxacIC*3*Rhj*0*oQFujHGpAkLdGOMSRMW*6*MYYkyYs*0*I*0*xJk*1*YoBtjTJqwQDfGmDRhgW*6*MMWnCAt*0*YY*9*KEBboxxqQJC*3*RjjEkTFujGGJMmfKmugDFm*8*EWjUSorK*2*ltbU*11*VUw/BYNBysrK*8*Pv*9*/X*6*MBboxo*0*BlZSU*5*OTlMmTIFEUl*1*dUwfVJUDBw*5*QWVnJ*1*KlT+/*0463*IxZhRobW*2*lsLDQwnyEEBEKCwsP+xOVBboxo*4*SF+chyJH*8*vC*3*RjjEkTFujGGJMmLNCNMUOitraWn/*3*sZ*4*f*9*uAsuuIDa*2*trDftw*111*zDU*089*ddiPG*8*nsKhdjRpl/+d+NbNpTP*6*DbPGFCLt/*61*Kxey*7*QF+pe+*9*KVOy*2*OxGD*5*fz*1*H*03*HPPDUgdRwNroRtjhsTSpUvZtm*0*bc+fO*5*eSTT+aMM*87*goosu*4*oQTTgDg*05*/+NPPnz*2*\n"  
 },  
 "metadata": {  
 "needs\_background": "light"  
 }  
 },  
 {  
 "output\_type": "stream",  
 "name": "stdout",  
 "text": [  
 "2 hidden layers with sigmoid - Test loss: 0.1095, Test accuracy: 0.9652*\n*"  
 ]  
 }  
 ]  
 }  
 ]  
}