SAI VARUN THABETI

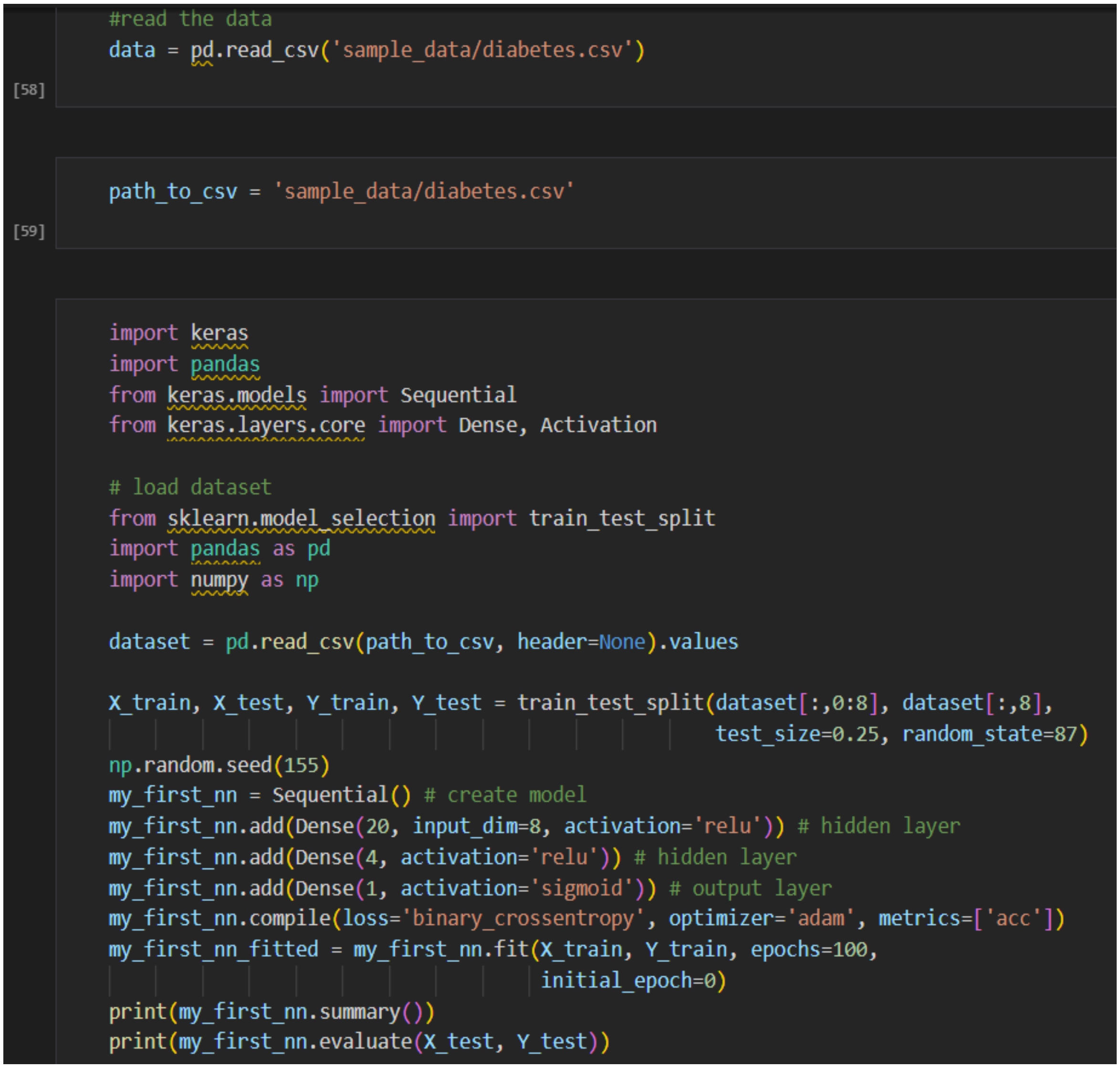
700741122

ASSIGNMENT 6

NEURAL NETWORKS AND DEEP LEARNING

Link for the recording: https://drive.google.com/file/d/16Zogw4-Ybm7-huAUZVk4cSxzcc5S55lB/view?usp=drive_link

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



```
#read the data
data = pd.read_csv('sample_data/diabetes.csv')
```

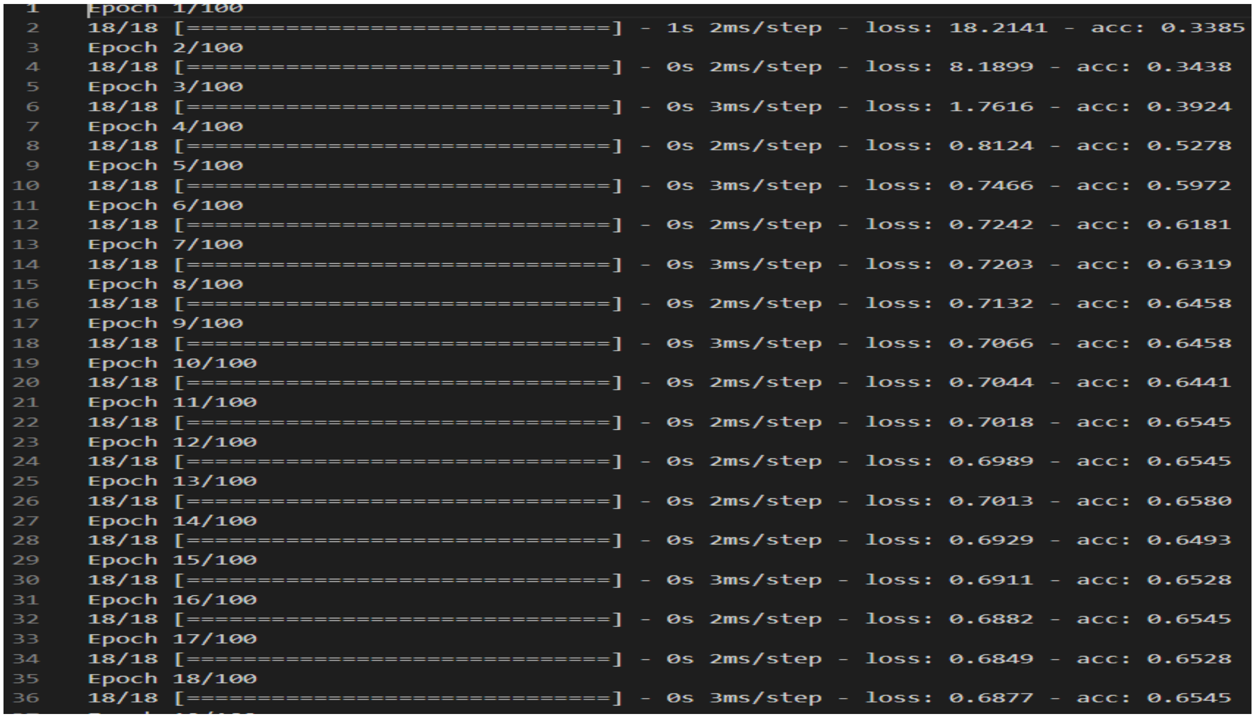
[58]

```
path_to_csv = 'sample_data/diabetes.csv'
```

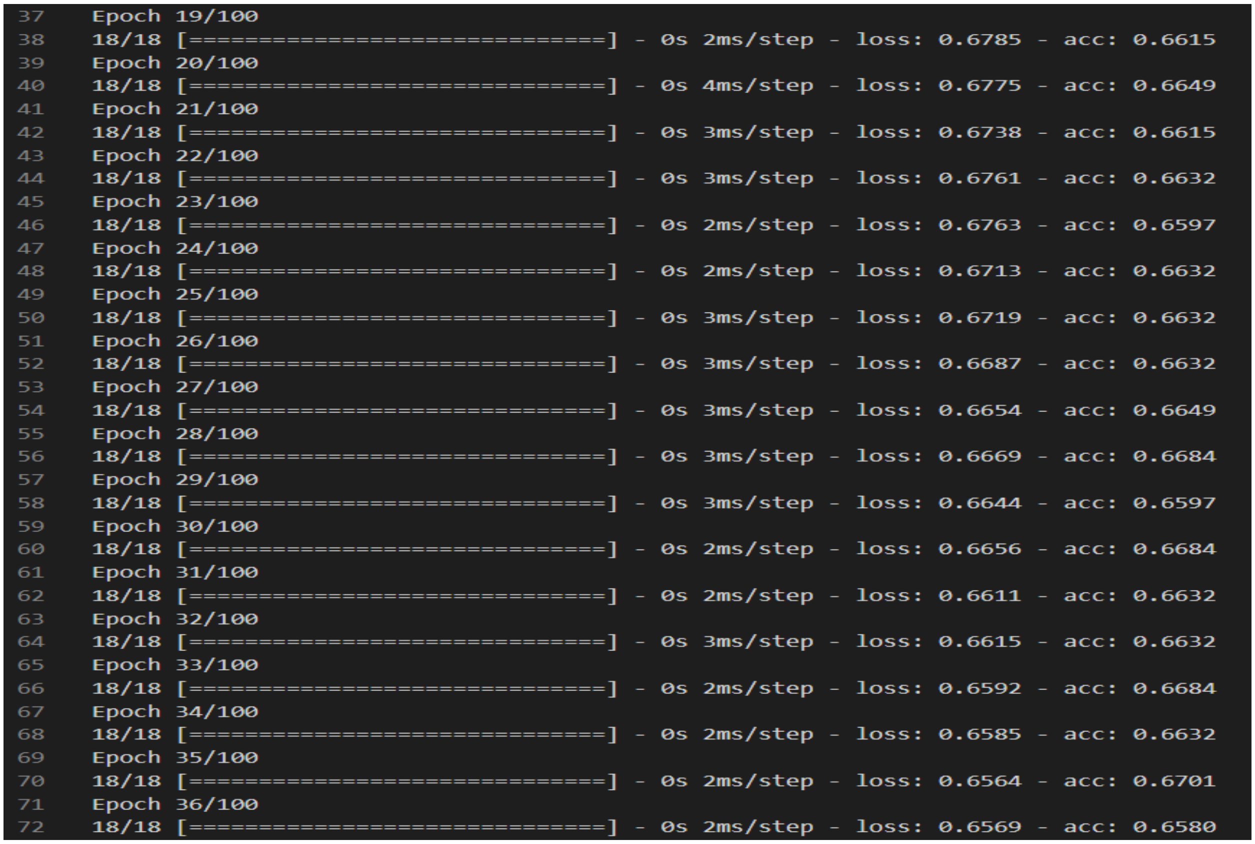
[59]

```
import keras
import pandas
from keras.models import Sequential
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))
```

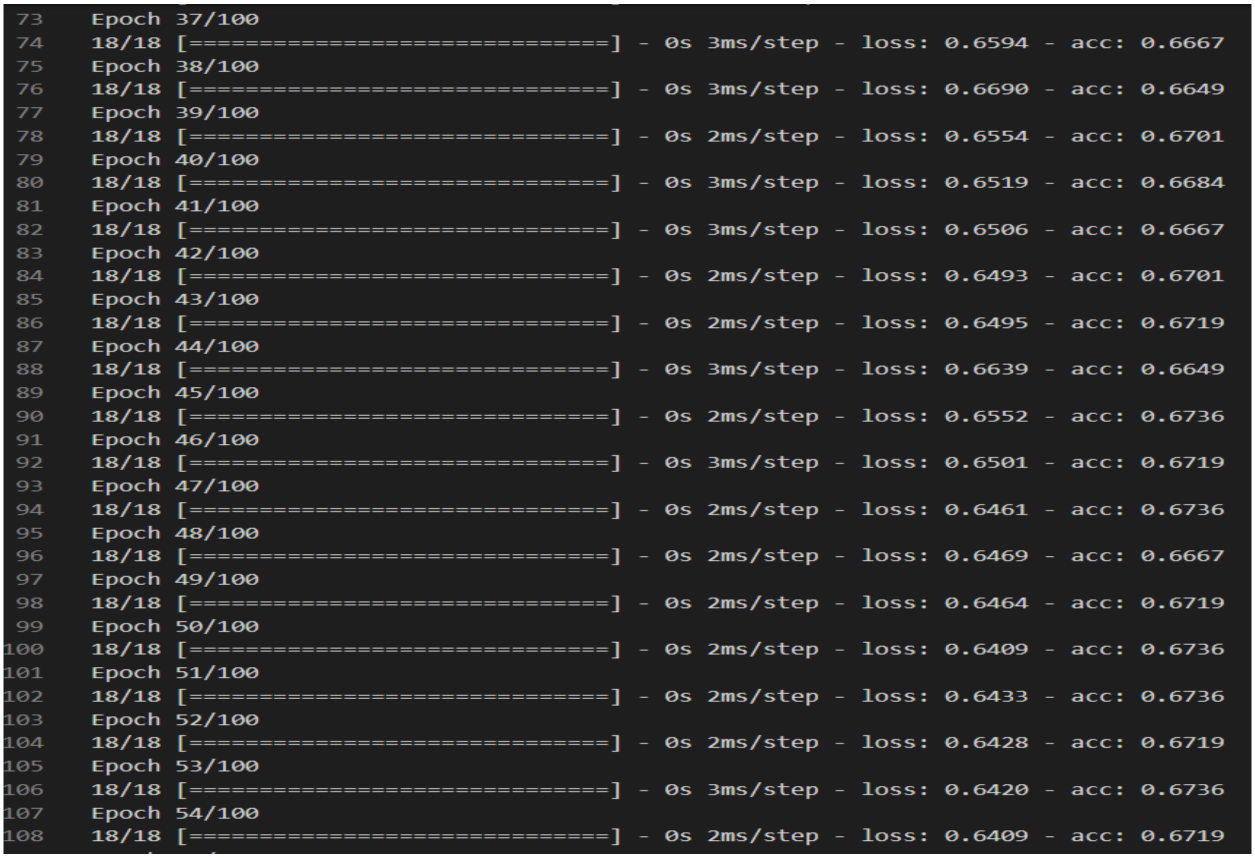
Output:



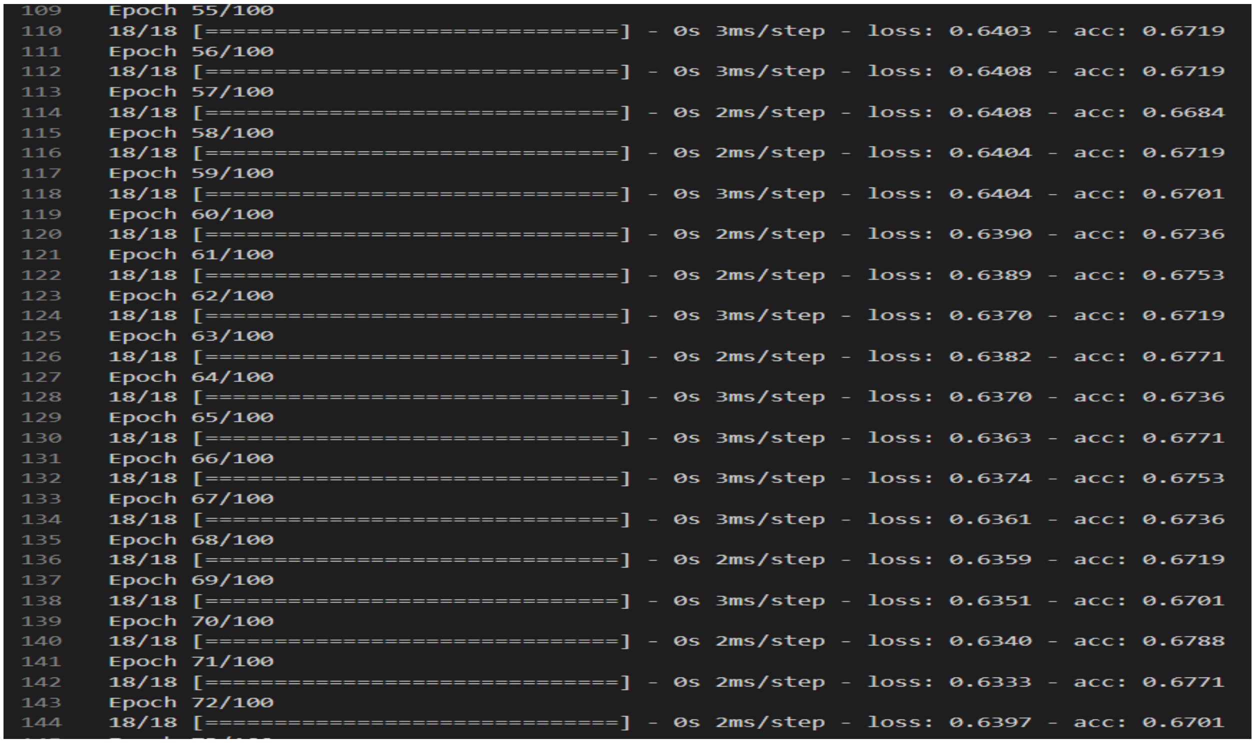
```
Epoch 1/100
18/18 [==============================] - 1s 2ms/step - loss: 18.2141 - acc: 0.3385
Epoch 2/100
18/18 [==============================] - 0s 2ms/step - loss: 8.1899 - acc: 0.3438
Epoch 3/100
18/18 [==============================] - 0s 3ms/step - loss: 1.7616 - acc: 0.3924
Epoch 4/100
18/18 [==============================] - 0s 2ms/step - loss: 0.8124 - acc: 0.5278
Epoch 5/100
18/18 [==============================] - 0s 3ms/step - loss: 0.7466 - acc: 0.5972
Epoch 6/100
18/18 [==============================] - 0s 2ms/step - loss: 0.7242 - acc: 0.6181
Epoch 7/100
18/18 [==============================] - 0s 3ms/step - loss: 0.7203 - acc: 0.6319
Epoch 8/100
18/18 [==============================] - 0s 2ms/step - loss: 0.7132 - acc: 0.6458
Epoch 9/100
18/18 [==============================] - 0s 3ms/step - loss: 0.7066 - acc: 0.6458
Epoch 10/100
18/18 [==============================] - 0s 2ms/step - loss: 0.7044 - acc: 0.6441
Epoch 11/100
18/18 [==============================] - 0s 2ms/step - loss: 0.7018 - acc: 0.6545
Epoch 12/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6989 - acc: 0.6545
Epoch 13/100
18/18 [==============================] - 0s 2ms/step - loss: 0.7013 - acc: 0.6580
Epoch 14/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6929 - acc: 0.6493
Epoch 15/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6911 - acc: 0.6528
Epoch 16/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6882 - acc: 0.6545
Epoch 17/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6849 - acc: 0.6528
Epoch 18/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6877 - acc: 0.6545
```



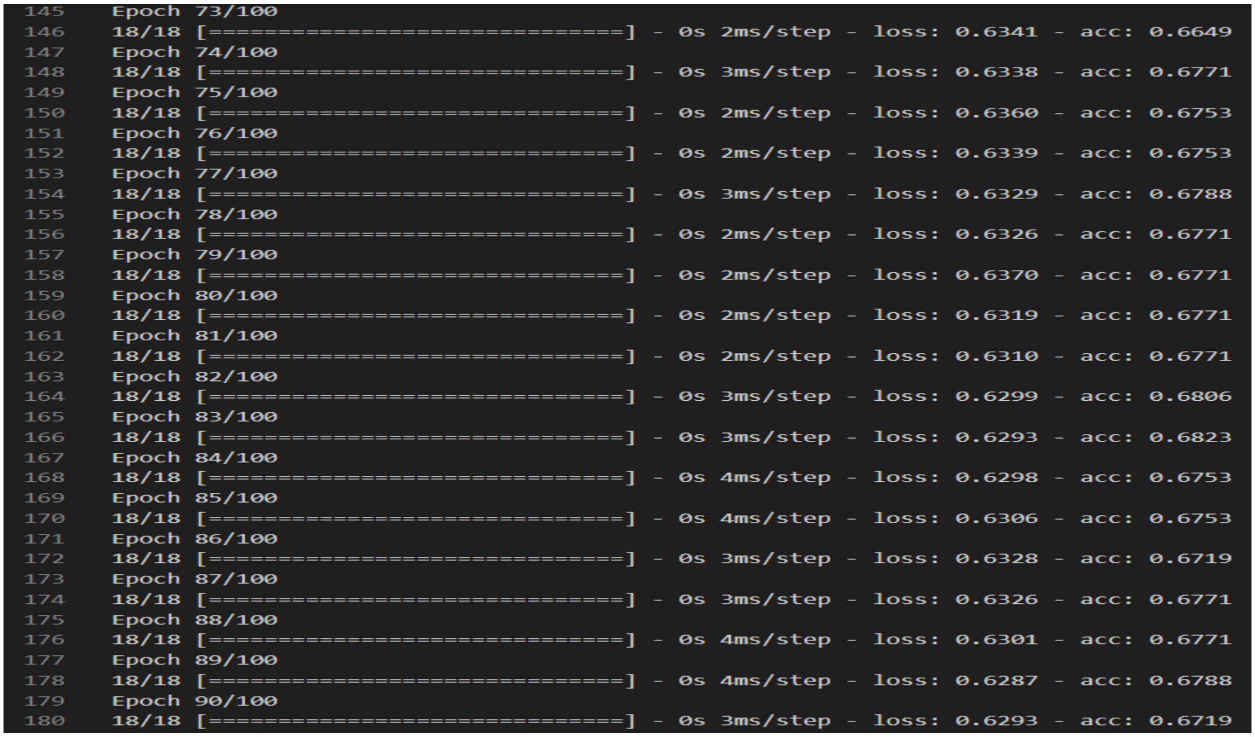
```
Epoch 19/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6785 - acc: 0.6615
Epoch 20/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6775 - acc: 0.6649
Epoch 21/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6738 - acc: 0.6615
Epoch 22/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6761 - acc: 0.6632
Epoch 23/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6763 - acc: 0.6597
Epoch 24/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6713 - acc: 0.6632
Epoch 25/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6719 - acc: 0.6632
Epoch 26/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6687 - acc: 0.6632
Epoch 27/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6654 - acc: 0.6649
Epoch 28/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6669 - acc: 0.6684
Epoch 29/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6644 - acc: 0.6597
Epoch 30/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6656 - acc: 0.6684
Epoch 31/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6611 - acc: 0.6632
Epoch 32/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6615 - acc: 0.6632
Epoch 33/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6592 - acc: 0.6684
Epoch 34/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6585 - acc: 0.6632
Epoch 35/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6564 - acc: 0.6701
Epoch 36/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6569 - acc: 0.6580
```



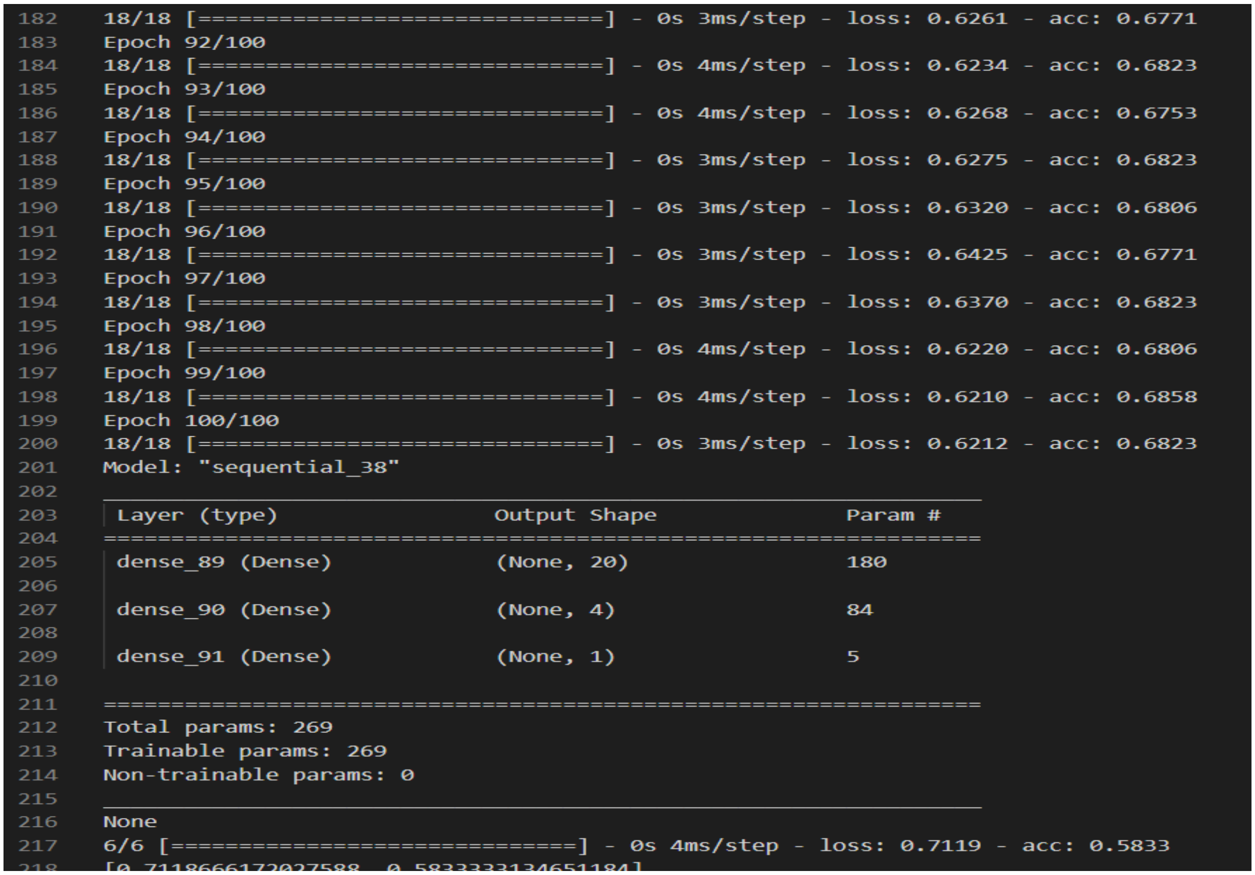
```
Epoch 37/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6594 - acc: 0.6667
Epoch 38/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6690 - acc: 0.6649
Epoch 39/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6554 - acc: 0.6701
Epoch 40/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6519 - acc: 0.6684
Epoch 41/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6506 - acc: 0.6667
Epoch 42/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6493 - acc: 0.6701
Epoch 43/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6495 - acc: 0.6719
Epoch 44/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6639 - acc: 0.6649
Epoch 45/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6552 - acc: 0.6736
Epoch 46/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6501 - acc: 0.6719
Epoch 47/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6461 - acc: 0.6736
Epoch 48/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6469 - acc: 0.6667
Epoch 49/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6464 - acc: 0.6719
Epoch 50/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6409 - acc: 0.6736
Epoch 51/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6433 - acc: 0.6736
Epoch 52/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6428 - acc: 0.6719
Epoch 53/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6420 - acc: 0.6736
Epoch 54/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6409 - acc: 0.6719
```



```
Epoch 55/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6403 - acc: 0.6719
Epoch 56/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6408 - acc: 0.6719
Epoch 57/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6408 - acc: 0.6684
Epoch 58/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6404 - acc: 0.6719
Epoch 59/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6404 - acc: 0.6701
Epoch 60/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6390 - acc: 0.6736
Epoch 61/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6389 - acc: 0.6753
Epoch 62/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6719
Epoch 63/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6382 - acc: 0.6771
Epoch 64/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6736
Epoch 65/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6363 - acc: 0.6771
Epoch 66/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6374 - acc: 0.6753
Epoch 67/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6361 - acc: 0.6736
Epoch 68/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6359 - acc: 0.6719
Epoch 69/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6351 - acc: 0.6701
Epoch 70/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6340 - acc: 0.6788
Epoch 71/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6333 - acc: 0.6771
Epoch 72/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6397 - acc: 0.6701
```

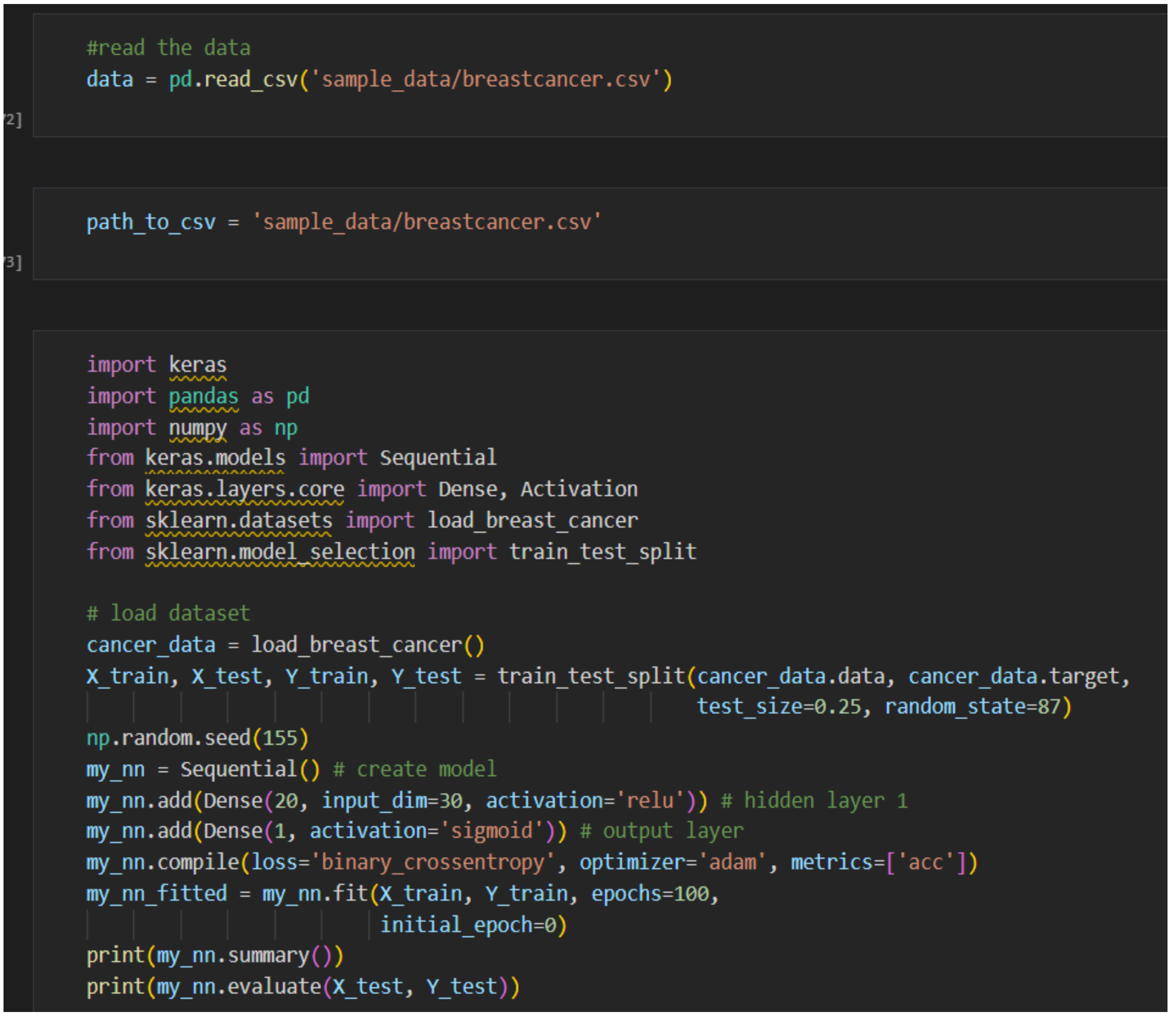


```
Epoch 73/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6341 - acc: 0.6649
Epoch 74/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6338 - acc: 0.6771
Epoch 75/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6360 - acc: 0.6753
Epoch 76/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6339 - acc: 0.6753
Epoch 77/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6329 - acc: 0.6788
Epoch 78/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6326 - acc: 0.6771
Epoch 79/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6370 - acc: 0.6771
Epoch 80/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6319 - acc: 0.6771
Epoch 81/100
18/18 [==============================] - 0s 2ms/step - loss: 0.6310 - acc: 0.6771
Epoch 82/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6299 - acc: 0.6806
Epoch 83/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6293 - acc: 0.6823
Epoch 84/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6298 - acc: 0.6753
Epoch 85/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6306 - acc: 0.6753
Epoch 86/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6328 - acc: 0.6719
Epoch 87/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6326 - acc: 0.6771
Epoch 88/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6301 - acc: 0.6771
Epoch 89/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6287 - acc: 0.6788
Epoch 90/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6293 - acc: 0.6719
```



```
18/18 [==============================] - 0s 3ms/step - loss: 0.6261 - acc: 0.6771
Epoch 92/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6234 - acc: 0.6823
Epoch 93/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6268 - acc: 0.6753
Epoch 94/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6275 - acc: 0.6823
Epoch 95/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6320 - acc: 0.6806
Epoch 96/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6425 - acc: 0.6771
Epoch 97/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6370 - acc: 0.6823
Epoch 98/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6220 - acc: 0.6806
Epoch 99/100
18/18 [==============================] - 0s 4ms/step - loss: 0.6210 - acc: 0.6858
Epoch 100/100
18/18 [==============================] - 0s 3ms/step - loss: 0.6212 - acc: 0.6823
Model: "sequential_38"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_89 (Dense)            (None, 20)                180

 dense_90 (Dense)            (None, 4)                 84

 dense_91 (Dense)            (None, 1)                 5

=================================================================
Total params: 269
Trainable params: 269
Non-trainable params: 0
_________________________________________________________________
None
6/6 [==============================] - 0s 4ms/step - loss: 0.7119 - acc: 0.5833
[0.7118666172027588, 0.5833333134651184]
```

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

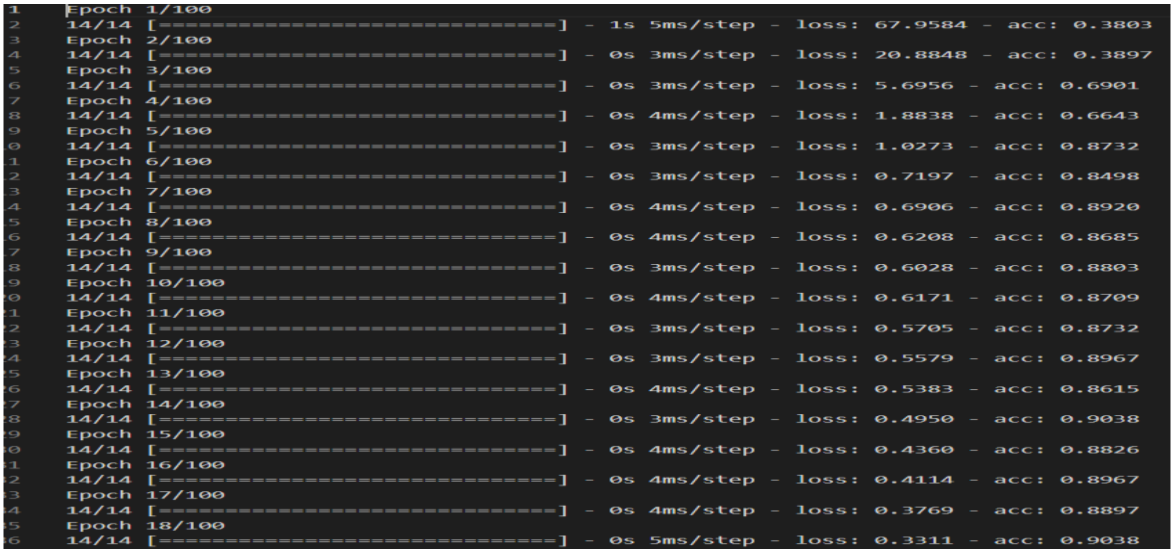


```python
#read the data
data = pd.read_csv('sample_data/breastcancer.csv')
```

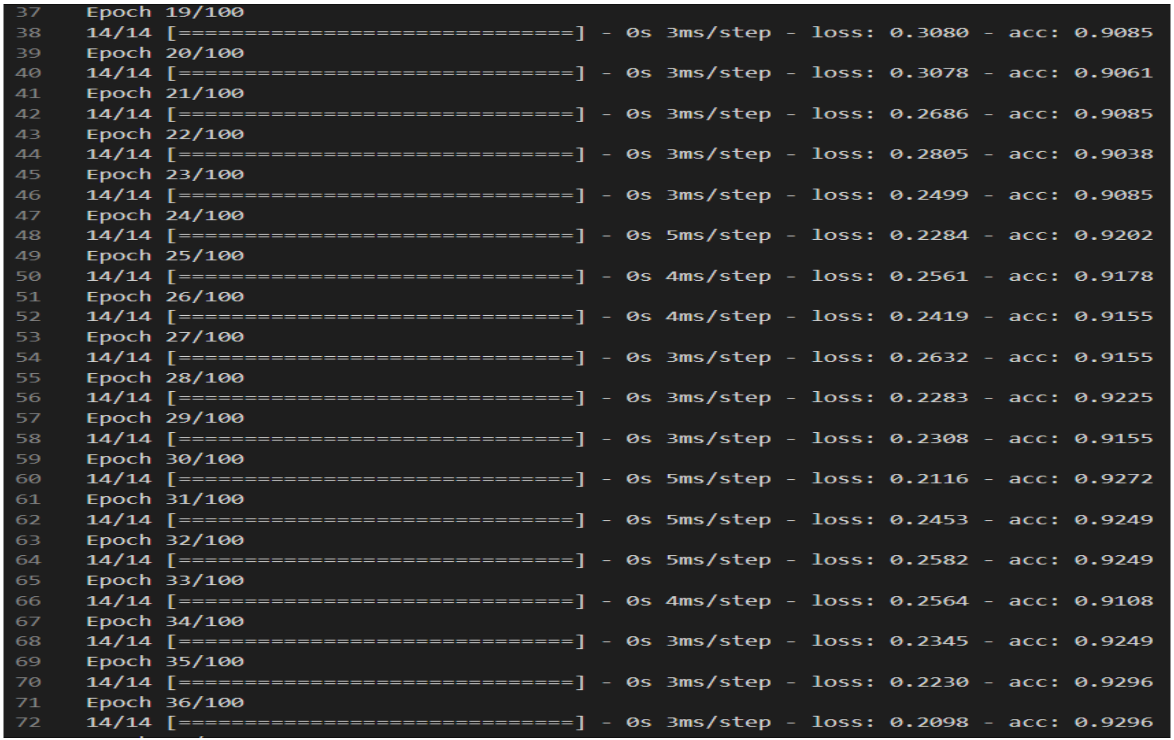
```python
path_to_csv = 'sample_data/breastcancer.csv'
```

```python
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers.core import Dense, Activation
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

# load dataset
cancer_data = load_breast_cancer()
X_train, X_test, Y_train, Y_test = train_test_split(cancer_data.data, cancer_data.target,
                                                    test_size=0.25, random_state=87)
np.random.seed(155)
my_nn = Sequential() # create model
my_nn.add(Dense(20, input_dim=30, activation='relu')) # hidden layer 1
my_nn.add(Dense(1, activation='sigmoid')) # output layer
my_nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
my_nn_fitted = my_nn.fit(X_train, Y_train, epochs=100,
                         initial_epoch=0)
print(my_nn.summary())
print(my_nn.evaluate(X_test, Y_test))
```

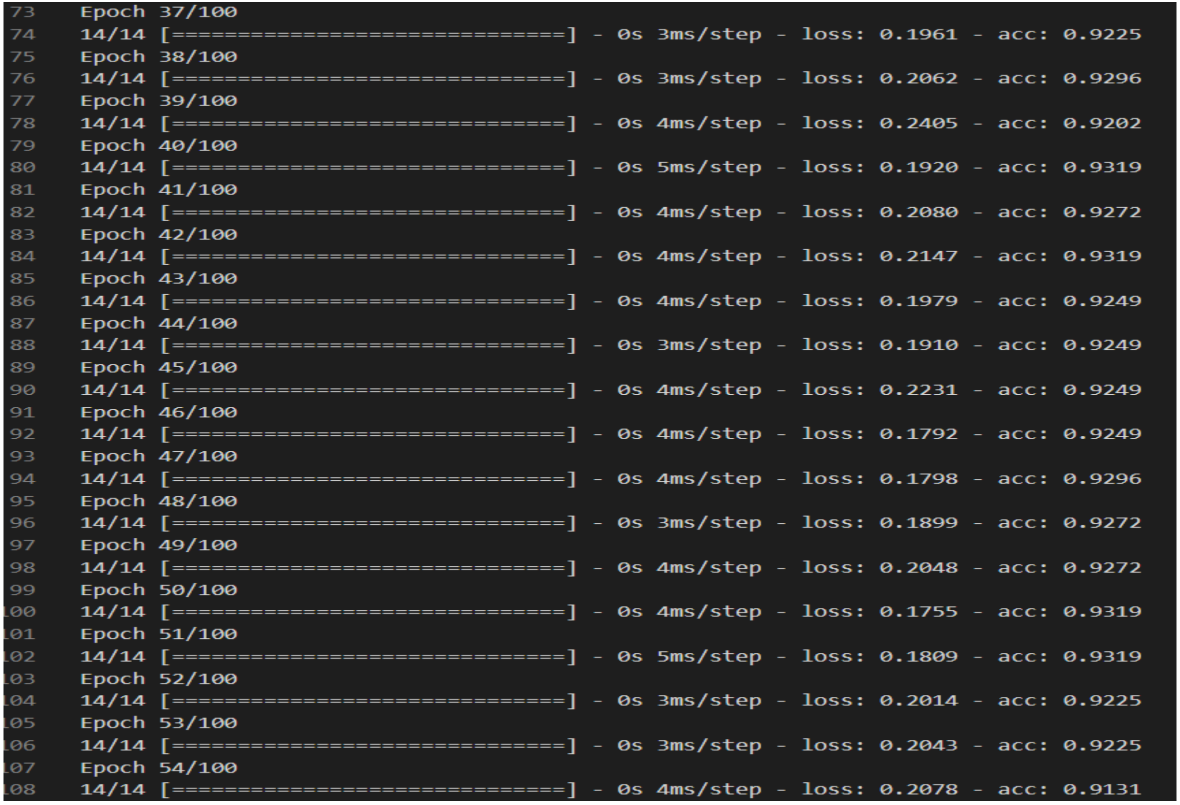
Output:



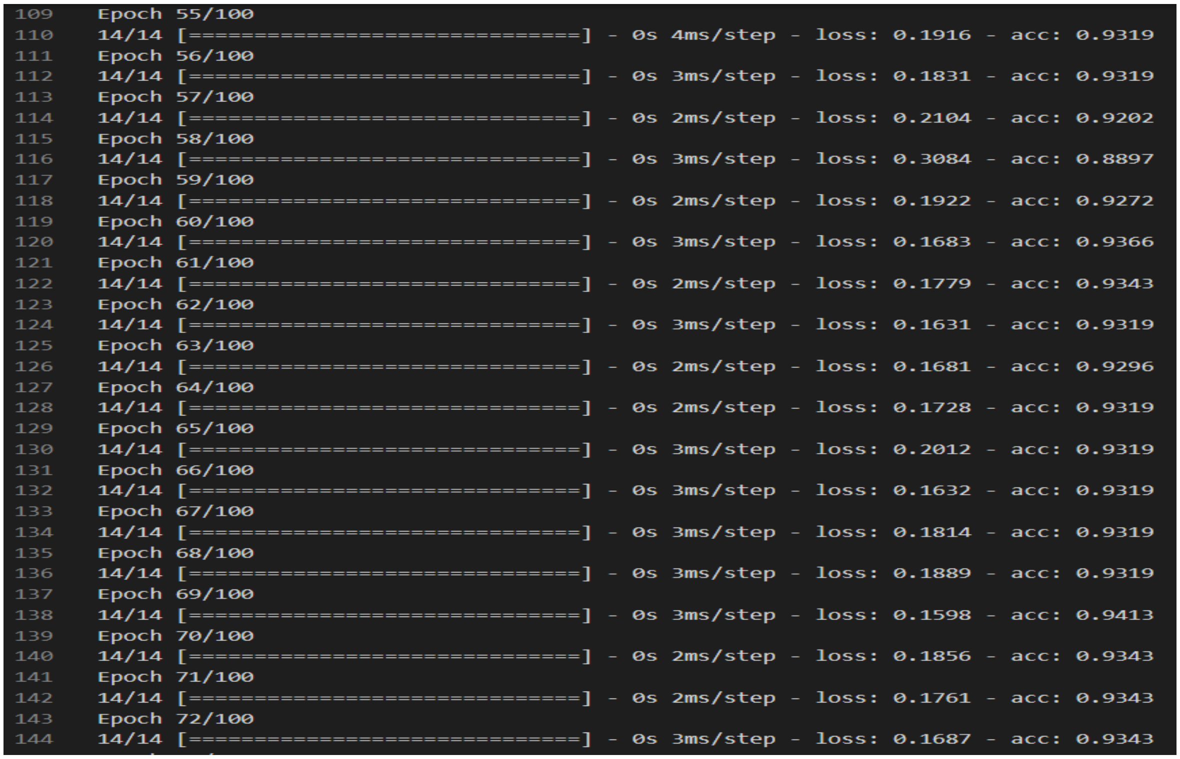
```
Epoch 1/100
14/14 [==============================] - 1s 5ms/step - loss: 67.9584 - acc: 0.3803
Epoch 2/100
14/14 [==============================] - 0s 3ms/step - loss: 20.8848 - acc: 0.3897
Epoch 3/100
14/14 [==============================] - 0s 3ms/step - loss: 5.6956 - acc: 0.6901
Epoch 4/100
14/14 [==============================] - 0s 4ms/step - loss: 1.8838 - acc: 0.6643
Epoch 5/100
14/14 [==============================] - 0s 3ms/step - loss: 1.0273 - acc: 0.8732
Epoch 6/100
14/14 [==============================] - 0s 3ms/step - loss: 0.7197 - acc: 0.8498
Epoch 7/100
14/14 [==============================] - 0s 4ms/step - loss: 0.6906 - acc: 0.8920
Epoch 8/100
14/14 [==============================] - 0s 4ms/step - loss: 0.6208 - acc: 0.8685
Epoch 9/100
14/14 [==============================] - 0s 3ms/step - loss: 0.6028 - acc: 0.8803
Epoch 10/100
14/14 [==============================] - 0s 4ms/step - loss: 0.6171 - acc: 0.8709
Epoch 11/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5705 - acc: 0.8732
Epoch 12/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5579 - acc: 0.8967
Epoch 13/100
14/14 [==============================] - 0s 4ms/step - loss: 0.5383 - acc: 0.8615
Epoch 14/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4950 - acc: 0.9038
Epoch 15/100
14/14 [==============================] - 0s 4ms/step - loss: 0.4360 - acc: 0.8826
Epoch 16/100
14/14 [==============================] - 0s 4ms/step - loss: 0.4114 - acc: 0.8967
Epoch 17/100
14/14 [==============================] - 0s 4ms/step - loss: 0.3769 - acc: 0.8897
Epoch 18/100
14/14 [==============================] - 0s 5ms/step - loss: 0.3311 - acc: 0.9038
```



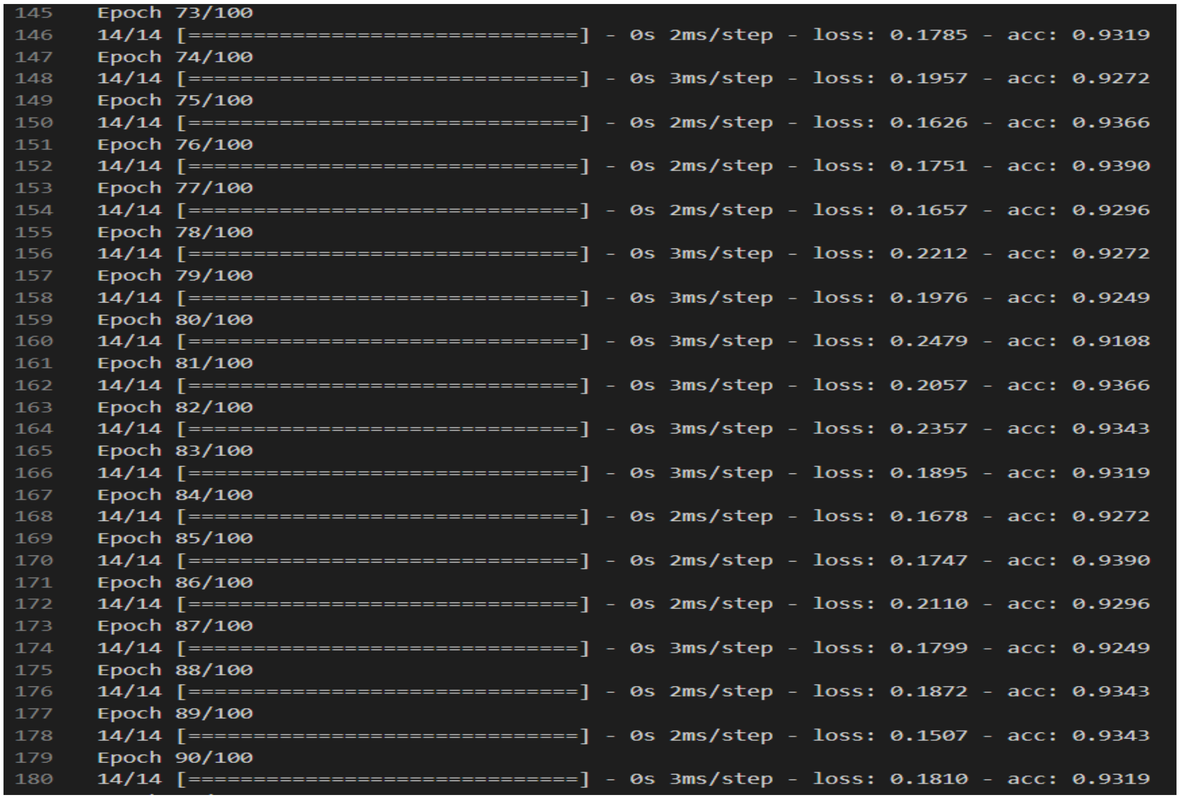
```
Epoch 19/100
14/14 [==============================] - 0s 3ms/step - loss: 0.3080 - acc: 0.9085
Epoch 20/100
14/14 [==============================] - 0s 3ms/step - loss: 0.3078 - acc: 0.9061
Epoch 21/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2686 - acc: 0.9085
Epoch 22/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2805 - acc: 0.9038
Epoch 23/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2499 - acc: 0.9085
Epoch 24/100
14/14 [==============================] - 0s 5ms/step - loss: 0.2284 - acc: 0.9202
Epoch 25/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2561 - acc: 0.9178
Epoch 26/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2419 - acc: 0.9155
Epoch 27/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2632 - acc: 0.9155
Epoch 28/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2283 - acc: 0.9225
Epoch 29/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2308 - acc: 0.9155
Epoch 30/100
14/14 [==============================] - 0s 5ms/step - loss: 0.2116 - acc: 0.9272
Epoch 31/100
14/14 [==============================] - 0s 5ms/step - loss: 0.2453 - acc: 0.9249
Epoch 32/100
14/14 [==============================] - 0s 5ms/step - loss: 0.2582 - acc: 0.9249
Epoch 33/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2564 - acc: 0.9108
Epoch 34/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2345 - acc: 0.9249
Epoch 35/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2230 - acc: 0.9296
Epoch 36/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2098 - acc: 0.9296
```



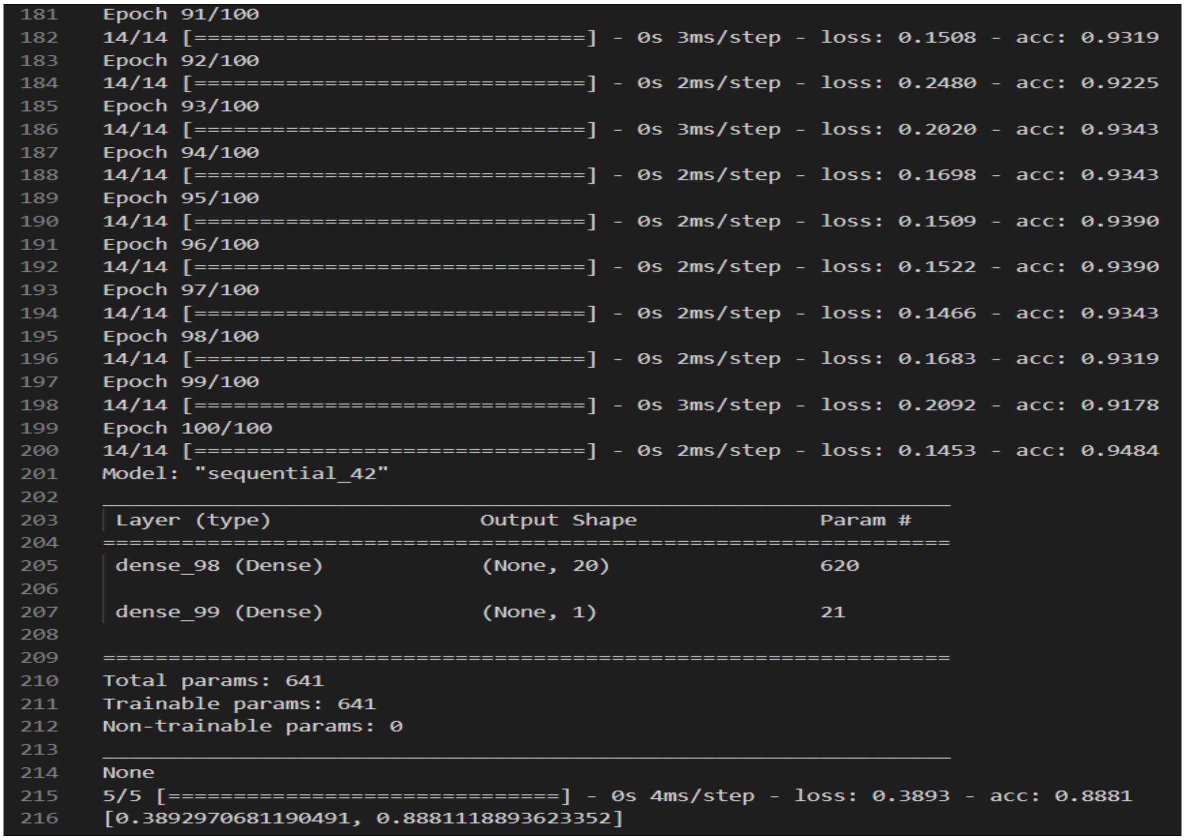
```
Epoch 37/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1961 - acc: 0.9225
Epoch 38/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2062 - acc: 0.9296
Epoch 39/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2405 - acc: 0.9202
Epoch 40/100
14/14 [==============================] - 0s 5ms/step - loss: 0.1920 - acc: 0.9319
Epoch 41/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2080 - acc: 0.9272
Epoch 42/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2147 - acc: 0.9319
Epoch 43/100
14/14 [==============================] - 0s 4ms/step - loss: 0.1979 - acc: 0.9249
Epoch 44/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1910 - acc: 0.9249
Epoch 45/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2231 - acc: 0.9249
Epoch 46/100
14/14 [==============================] - 0s 4ms/step - loss: 0.1792 - acc: 0.9249
Epoch 47/100
14/14 [==============================] - 0s 4ms/step - loss: 0.1798 - acc: 0.9296
Epoch 48/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1899 - acc: 0.9272
Epoch 49/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2048 - acc: 0.9272
Epoch 50/100
14/14 [==============================] - 0s 4ms/step - loss: 0.1755 - acc: 0.9319
Epoch 51/100
14/14 [==============================] - 0s 5ms/step - loss: 0.1809 - acc: 0.9319
Epoch 52/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2014 - acc: 0.9225
Epoch 53/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2043 - acc: 0.9225
Epoch 54/100
14/14 [==============================] - 0s 4ms/step - loss: 0.2078 - acc: 0.9131
```



```
Epoch 55/100
14/14 [==============================] - 0s 4ms/step - loss: 0.1916 - acc: 0.9319
Epoch 56/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1831 - acc: 0.9319
Epoch 57/100
14/14 [==============================] - 0s 2ms/step - loss: 0.2104 - acc: 0.9202
Epoch 58/100
14/14 [==============================] - 0s 3ms/step - loss: 0.3084 - acc: 0.8897
Epoch 59/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1922 - acc: 0.9272
Epoch 60/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1683 - acc: 0.9366
Epoch 61/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1779 - acc: 0.9343
Epoch 62/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1631 - acc: 0.9319
Epoch 63/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1681 - acc: 0.9296
Epoch 64/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1728 - acc: 0.9319
Epoch 65/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2012 - acc: 0.9319
Epoch 66/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1632 - acc: 0.9319
Epoch 67/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1814 - acc: 0.9319
Epoch 68/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1889 - acc: 0.9319
Epoch 69/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1598 - acc: 0.9413
Epoch 70/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1856 - acc: 0.9343
Epoch 71/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1761 - acc: 0.9343
Epoch 72/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1687 - acc: 0.9343
```

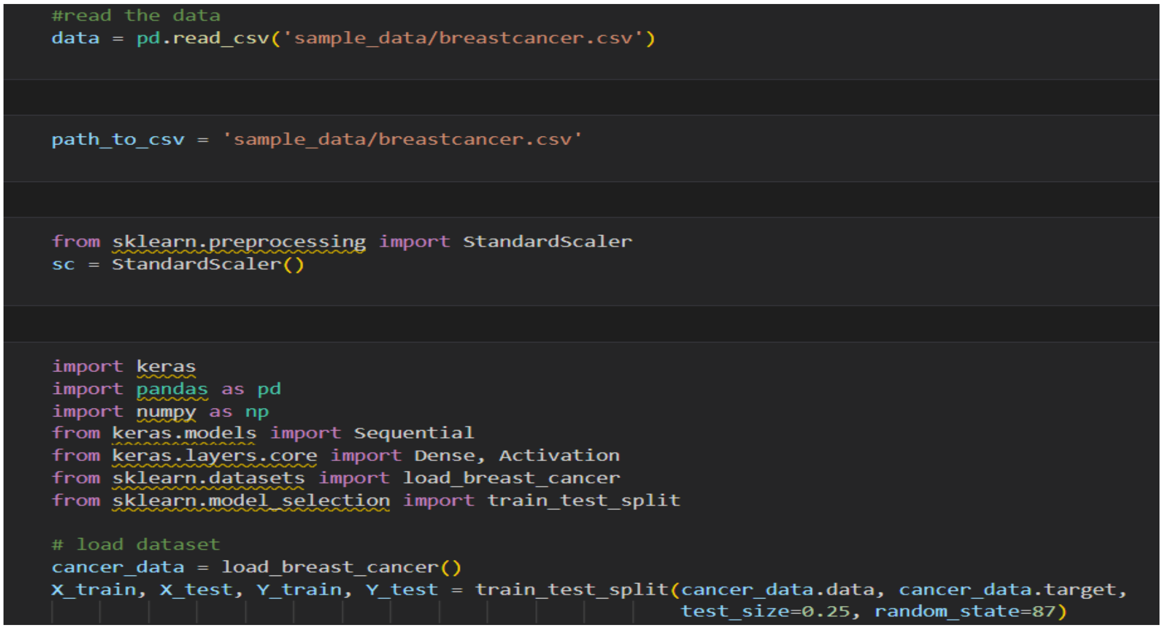


```
Epoch 73/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1785 - acc: 0.9319
Epoch 74/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1957 - acc: 0.9272
Epoch 75/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1626 - acc: 0.9366
Epoch 76/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1751 - acc: 0.9390
Epoch 77/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1657 - acc: 0.9296
Epoch 78/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2212 - acc: 0.9272
Epoch 79/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1976 - acc: 0.9249
Epoch 80/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2479 - acc: 0.9108
Epoch 81/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2057 - acc: 0.9366
Epoch 82/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2357 - acc: 0.9343
Epoch 83/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1895 - acc: 0.9319
Epoch 84/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1678 - acc: 0.9272
Epoch 85/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1747 - acc: 0.9390
Epoch 86/100
14/14 [==============================] - 0s 2ms/step - loss: 0.2110 - acc: 0.9296
Epoch 87/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1799 - acc: 0.9249
Epoch 88/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1872 - acc: 0.9343
Epoch 89/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1507 - acc: 0.9343
Epoch 90/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1810 - acc: 0.9319
```



```
Epoch 91/100
14/14 [==============================] - 0s 3ms/step - loss: 0.1508 - acc: 0.9319
Epoch 92/100
14/14 [==============================] - 0s 2ms/step - loss: 0.2480 - acc: 0.9225
Epoch 93/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2020 - acc: 0.9343
Epoch 94/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1698 - acc: 0.9343
Epoch 95/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1509 - acc: 0.9390
Epoch 96/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1522 - acc: 0.9390
Epoch 97/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1466 - acc: 0.9343
Epoch 98/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1683 - acc: 0.9319
Epoch 99/100
14/14 [==============================] - 0s 3ms/step - loss: 0.2092 - acc: 0.9178
Epoch 100/100
14/14 [==============================] - 0s 2ms/step - loss: 0.1453 - acc: 0.9484
Model: "sequential_42"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_98 (Dense)            (None, 20)                620

 dense_99 (Dense)            (None, 1)                 21

=================================================================
Total params: 641
Trainable params: 641
Non-trainable params: 0
_________________________________________________________________
None
5/5 [==============================] - 0s 4ms/step - loss: 0.3893 - acc: 0.8881
[0.3892970681190491, 0.8881118893623352]
```

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() Breast Cancer dataset is designated to predict if a patient has Malignant (M) or Benign = B cancer

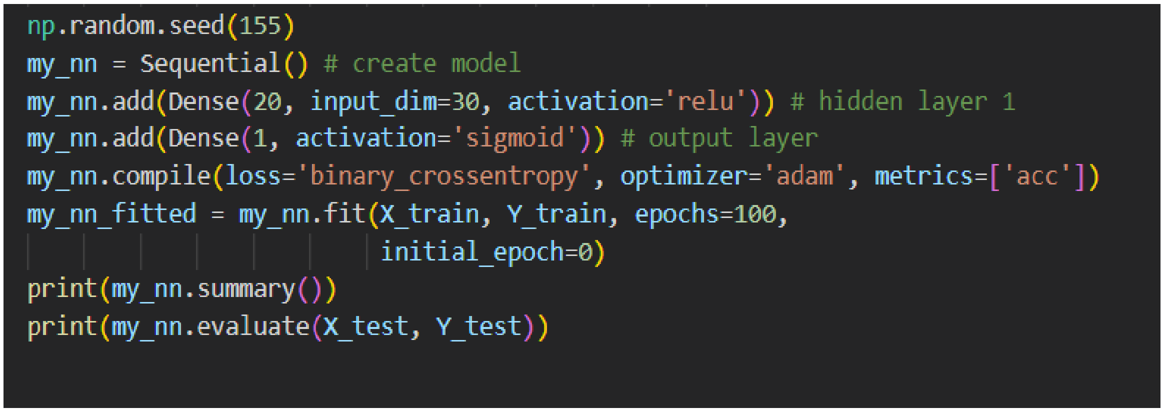


```
#read the data
data = pd.read_csv('sample_data/breastcancer.csv')
```

```
path_to_csv = 'sample_data/breastcancer.csv'
```

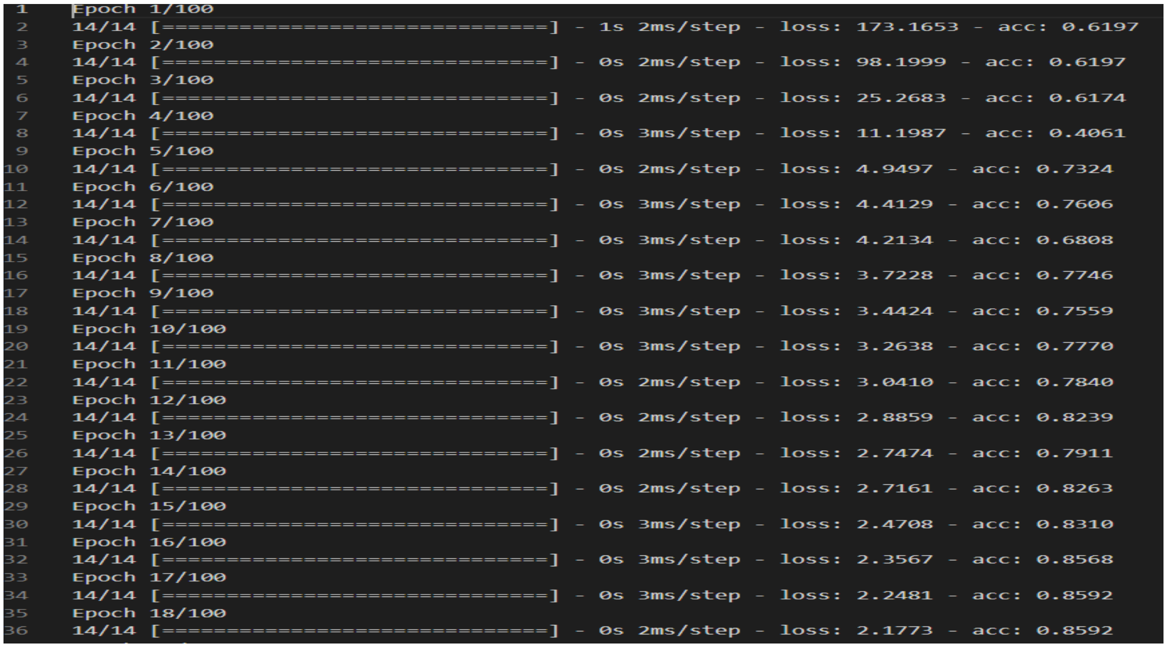
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

```
import keras
import pandas as pd
import numpy as np
from keras.models import Sequential
from keras.layers.core import Dense, Activation
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split

# load dataset
cancer_data = load_breast_cancer()
X_train, X_test, Y_train, Y_test = train_test_split(cancer_data.data, cancer_data.target,
                                                    test_size=0.25, random_state=87)
```

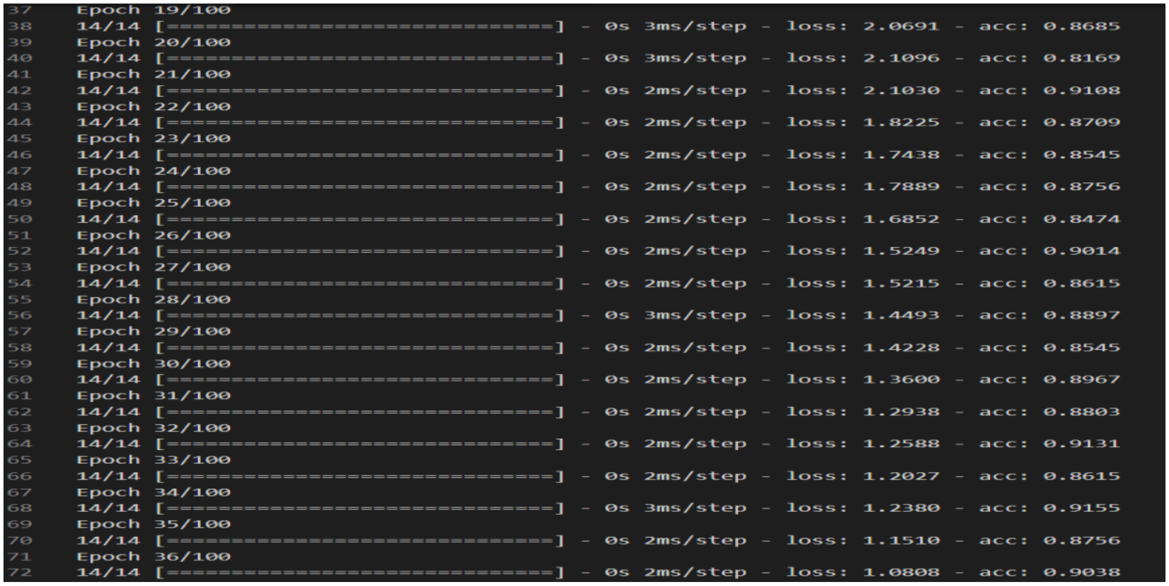


```
np.random.seed(155)
my_nn = Sequential() # create model
my_nn.add(Dense(20, input_dim=30, activation='relu')) # hidden layer 1
my_nn.add(Dense(1, activation='sigmoid')) # output layer
my_nn.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
my_nn_fitted = my_nn.fit(X_train, Y_train, epochs=100,
                         initial_epoch=0)
print(my_nn.summary())
print(my_nn.evaluate(X_test, Y_test))
```

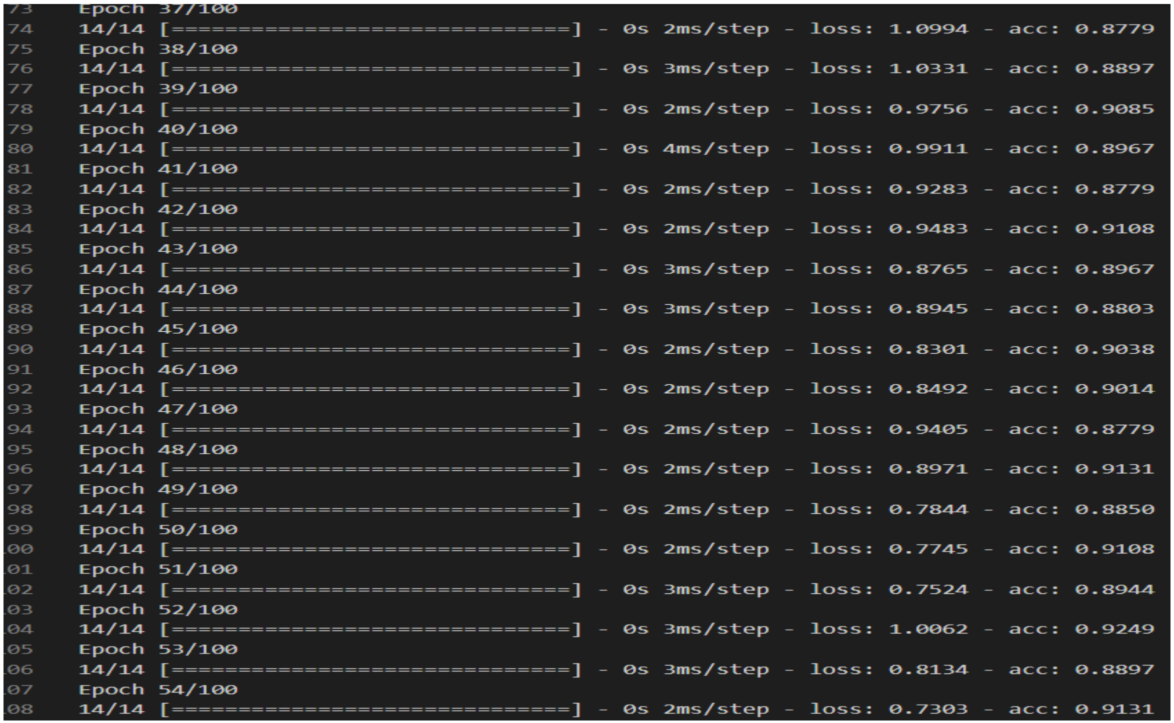
Output:



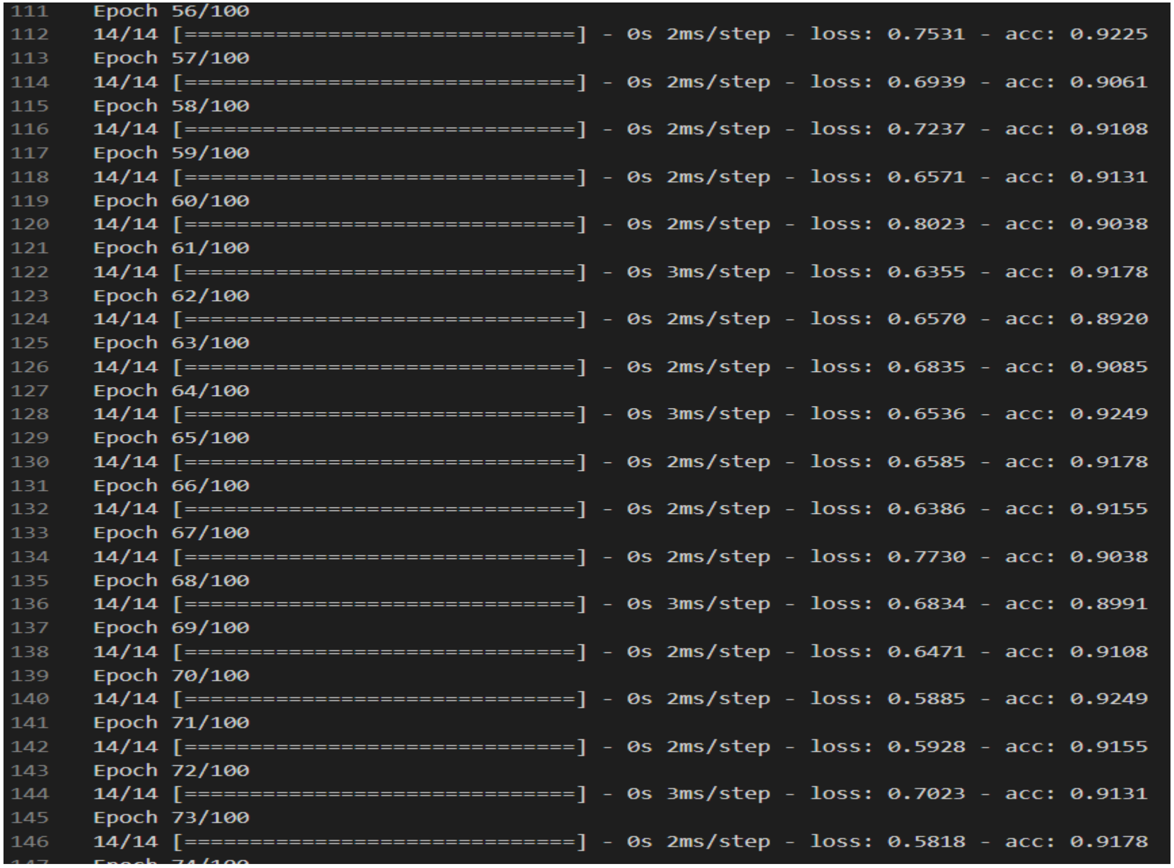
```
Epoch 1/100
14/14 [==============================] - 1s 2ms/step - loss: 173.1653 - acc: 0.6197
Epoch 2/100
14/14 [==============================] - 0s 2ms/step - loss: 98.1999 - acc: 0.6197
Epoch 3/100
14/14 [==============================] - 0s 2ms/step - loss: 25.2683 - acc: 0.6174
Epoch 4/100
14/14 [==============================] - 0s 3ms/step - loss: 11.1987 - acc: 0.4061
Epoch 5/100
14/14 [==============================] - 0s 2ms/step - loss: 4.9497 - acc: 0.7324
Epoch 6/100
14/14 [==============================] - 0s 3ms/step - loss: 4.4129 - acc: 0.7606
Epoch 7/100
14/14 [==============================] - 0s 3ms/step - loss: 4.2134 - acc: 0.6808
Epoch 8/100
14/14 [==============================] - 0s 3ms/step - loss: 3.7228 - acc: 0.7746
Epoch 9/100
14/14 [==============================] - 0s 3ms/step - loss: 3.4424 - acc: 0.7559
Epoch 10/100
14/14 [==============================] - 0s 3ms/step - loss: 3.2638 - acc: 0.7770
Epoch 11/100
14/14 [==============================] - 0s 2ms/step - loss: 3.0410 - acc: 0.7840
Epoch 12/100
14/14 [==============================] - 0s 2ms/step - loss: 2.8859 - acc: 0.8239
Epoch 13/100
14/14 [==============================] - 0s 2ms/step - loss: 2.7474 - acc: 0.7911
Epoch 14/100
14/14 [==============================] - 0s 2ms/step - loss: 2.7161 - acc: 0.8263
Epoch 15/100
14/14 [==============================] - 0s 3ms/step - loss: 2.4708 - acc: 0.8310
Epoch 16/100
14/14 [==============================] - 0s 3ms/step - loss: 2.3567 - acc: 0.8568
Epoch 17/100
14/14 [==============================] - 0s 3ms/step - loss: 2.2481 - acc: 0.8592
Epoch 18/100
14/14 [==============================] - 0s 2ms/step - loss: 2.1773 - acc: 0.8592
```



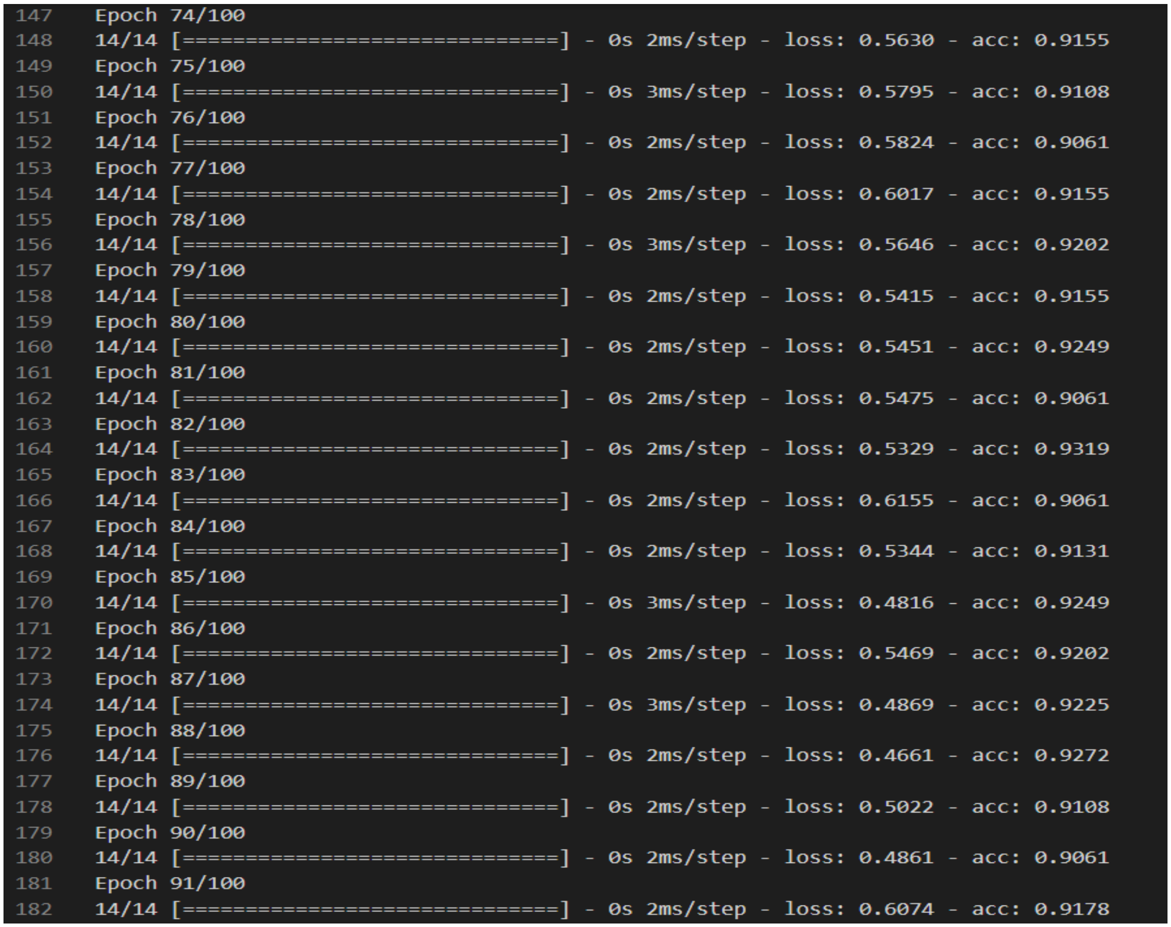
```
Epoch 19/100
14/14 [==============================] - 0s 3ms/step - loss: 2.0691 - acc: 0.8685
Epoch 20/100
14/14 [==============================] - 0s 3ms/step - loss: 2.1096 - acc: 0.8169
Epoch 21/100
14/14 [==============================] - 0s 2ms/step - loss: 2.1030 - acc: 0.9108
Epoch 22/100
14/14 [==============================] - 0s 2ms/step - loss: 1.8225 - acc: 0.8709
Epoch 23/100
14/14 [==============================] - 0s 2ms/step - loss: 1.7438 - acc: 0.8545
Epoch 24/100
14/14 [==============================] - 0s 2ms/step - loss: 1.7889 - acc: 0.8756
Epoch 25/100
14/14 [==============================] - 0s 2ms/step - loss: 1.6852 - acc: 0.8474
Epoch 26/100
14/14 [==============================] - 0s 2ms/step - loss: 1.5249 - acc: 0.9014
Epoch 27/100
14/14 [==============================] - 0s 2ms/step - loss: 1.5215 - acc: 0.8615
Epoch 28/100
14/14 [==============================] - 0s 3ms/step - loss: 1.4493 - acc: 0.8897
Epoch 29/100
14/14 [==============================] - 0s 2ms/step - loss: 1.4228 - acc: 0.8545
Epoch 30/100
14/14 [==============================] - 0s 2ms/step - loss: 1.3600 - acc: 0.8967
Epoch 31/100
14/14 [==============================] - 0s 2ms/step - loss: 1.2938 - acc: 0.8803
Epoch 32/100
14/14 [==============================] - 0s 2ms/step - loss: 1.2588 - acc: 0.9131
Epoch 33/100
14/14 [==============================] - 0s 2ms/step - loss: 1.2027 - acc: 0.8615
Epoch 34/100
14/14 [==============================] - 0s 3ms/step - loss: 1.2380 - acc: 0.9155
Epoch 35/100
14/14 [==============================] - 0s 2ms/step - loss: 1.1510 - acc: 0.8756
Epoch 36/100
14/14 [==============================] - 0s 2ms/step - loss: 1.0808 - acc: 0.9038
```



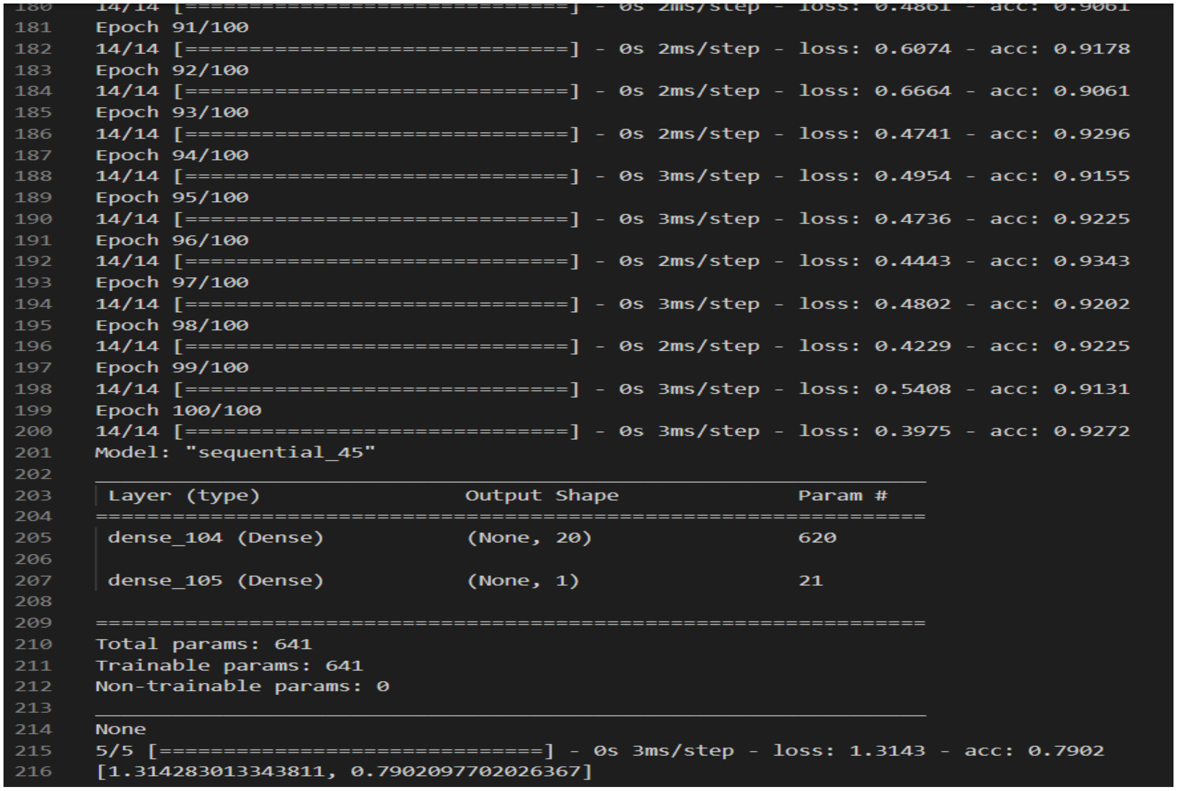
```
Epoch 37/100
14/14 [==============================] - 0s 2ms/step - loss: 1.0994 - acc: 0.8779
Epoch 38/100
14/14 [==============================] - 0s 3ms/step - loss: 1.0331 - acc: 0.8897
Epoch 39/100
14/14 [==============================] - 0s 2ms/step - loss: 0.9756 - acc: 0.9085
Epoch 40/100
14/14 [==============================] - 0s 4ms/step - loss: 0.9911 - acc: 0.8967
Epoch 41/100
14/14 [==============================] - 0s 2ms/step - loss: 0.9283 - acc: 0.8779
Epoch 42/100
14/14 [==============================] - 0s 2ms/step - loss: 0.9483 - acc: 0.9108
Epoch 43/100
14/14 [==============================] - 0s 3ms/step - loss: 0.8765 - acc: 0.8967
Epoch 44/100
14/14 [==============================] - 0s 3ms/step - loss: 0.8945 - acc: 0.8803
Epoch 45/100
14/14 [==============================] - 0s 2ms/step - loss: 0.8301 - acc: 0.9038
Epoch 46/100
14/14 [==============================] - 0s 2ms/step - loss: 0.8492 - acc: 0.9014
Epoch 47/100
14/14 [==============================] - 0s 2ms/step - loss: 0.9405 - acc: 0.8779
Epoch 48/100
14/14 [==============================] - 0s 2ms/step - loss: 0.8971 - acc: 0.9131
Epoch 49/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7844 - acc: 0.8850
Epoch 50/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7745 - acc: 0.9108
Epoch 51/100
14/14 [==============================] - 0s 3ms/step - loss: 0.7524 - acc: 0.8944
Epoch 52/100
14/14 [==============================] - 0s 3ms/step - loss: 1.0062 - acc: 0.9249
Epoch 53/100
14/14 [==============================] - 0s 3ms/step - loss: 0.8134 - acc: 0.8897
Epoch 54/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7303 - acc: 0.9131
```



```
Epoch 56/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7531 - acc: 0.9225
Epoch 57/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6939 - acc: 0.9061
Epoch 58/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7237 - acc: 0.9108
Epoch 59/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6571 - acc: 0.9131
Epoch 60/100
14/14 [==============================] - 0s 2ms/step - loss: 0.8023 - acc: 0.9038
Epoch 61/100
14/14 [==============================] - 0s 3ms/step - loss: 0.6355 - acc: 0.9178
Epoch 62/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6570 - acc: 0.8920
Epoch 63/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6835 - acc: 0.9085
Epoch 64/100
14/14 [==============================] - 0s 3ms/step - loss: 0.6536 - acc: 0.9249
Epoch 65/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6585 - acc: 0.9178
Epoch 66/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6386 - acc: 0.9155
Epoch 67/100
14/14 [==============================] - 0s 2ms/step - loss: 0.7730 - acc: 0.9038
Epoch 68/100
14/14 [==============================] - 0s 3ms/step - loss: 0.6834 - acc: 0.8991
Epoch 69/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6471 - acc: 0.9108
Epoch 70/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5885 - acc: 0.9249
Epoch 71/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5928 - acc: 0.9155
Epoch 72/100
14/14 [==============================] - 0s 3ms/step - loss: 0.7023 - acc: 0.9131
Epoch 73/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5818 - acc: 0.9178
Epoch 74/100
```



```
Epoch 74/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5630 - acc: 0.9155
Epoch 75/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5795 - acc: 0.9108
Epoch 76/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5824 - acc: 0.9061
Epoch 77/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6017 - acc: 0.9155
Epoch 78/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5646 - acc: 0.9202
Epoch 79/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5415 - acc: 0.9155
Epoch 80/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5451 - acc: 0.9249
Epoch 81/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5475 - acc: 0.9061
Epoch 82/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5329 - acc: 0.9319
Epoch 83/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6155 - acc: 0.9061
Epoch 84/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5344 - acc: 0.9131
Epoch 85/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4816 - acc: 0.9249
Epoch 86/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5469 - acc: 0.9202
Epoch 87/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4869 - acc: 0.9225
Epoch 88/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4661 - acc: 0.9272
Epoch 89/100
14/14 [==============================] - 0s 2ms/step - loss: 0.5022 - acc: 0.9108
Epoch 90/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4861 - acc: 0.9061
Epoch 91/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6074 - acc: 0.9178
```

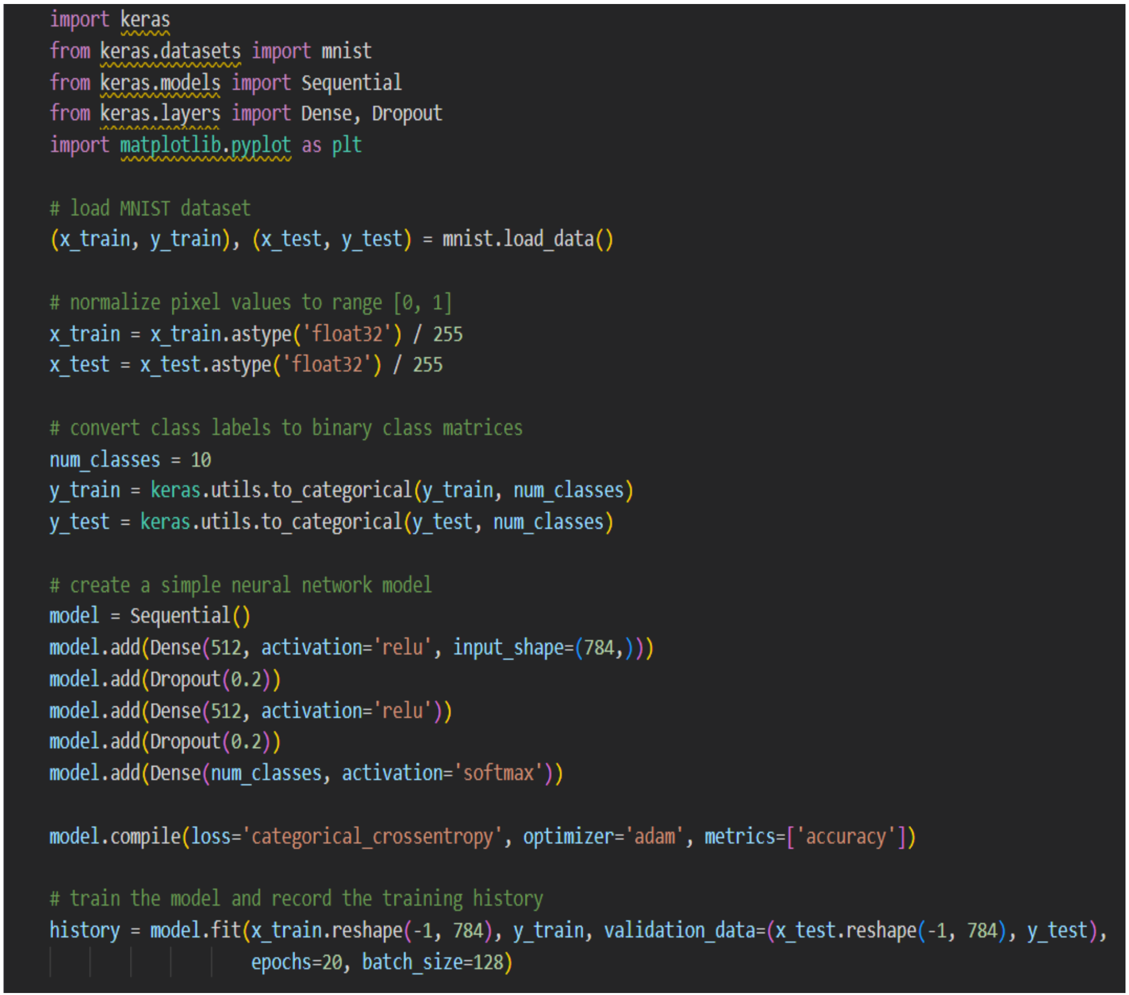


```
14/14 [==============================] - 0s 2ms/step - loss: 0.4861 - acc: 0.9061
Epoch 91/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6074 - acc: 0.9178
```

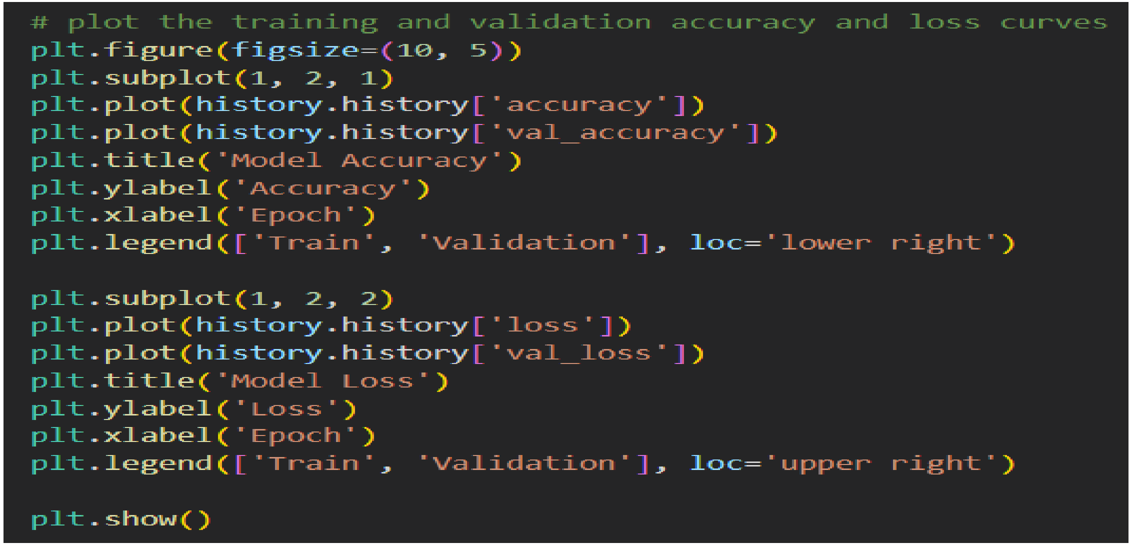
```
Epoch 92/100
14/14 [==============================] - 0s 2ms/step - loss: 0.6664 - acc: 0.9061
Epoch 93/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4741 - acc: 0.9296
Epoch 94/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4954 - acc: 0.9155
Epoch 95/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4736 - acc: 0.9225
Epoch 96/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4443 - acc: 0.9343
Epoch 97/100
14/14 [==============================] - 0s 3ms/step - loss: 0.4802 - acc: 0.9202
Epoch 98/100
14/14 [==============================] - 0s 2ms/step - loss: 0.4229 - acc: 0.9225
Epoch 99/100
14/14 [==============================] - 0s 3ms/step - loss: 0.5408 - acc: 0.9131
Epoch 100/100
14/14 [==============================] - 0s 3ms/step - loss: 0.3975 - acc: 0.9272
Model: "sequential_45"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_104 (Dense)           (None, 20)                620

 dense_105 (Dense)           (None, 1)                 21

=================================================================
Total params: 641
Trainable params: 641
Non-trainable params: 0
_________________________________________________________________
None
5/5 [==============================] - 0s 3ms/step - loss: 1.3143 - acc: 0.7902
[1.314283013343811, 0.7902097702026367]
```

Use Image Classification on the hand written digits data set (mnist)

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

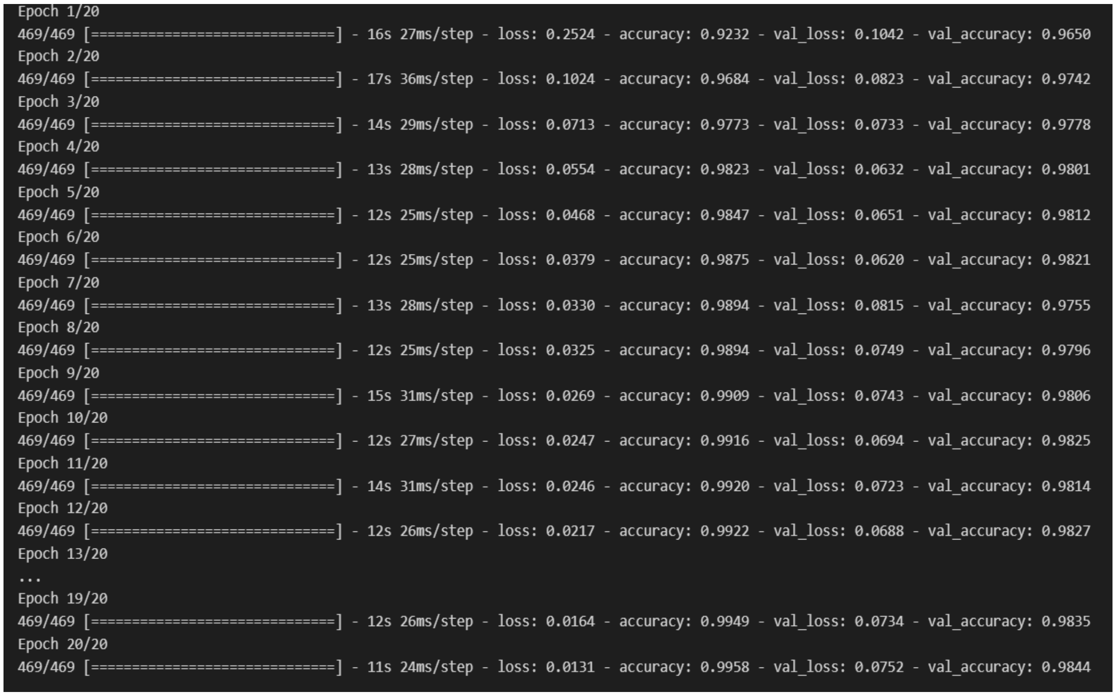


```
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout
import matplotlib.pyplot as plt

# 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)
```

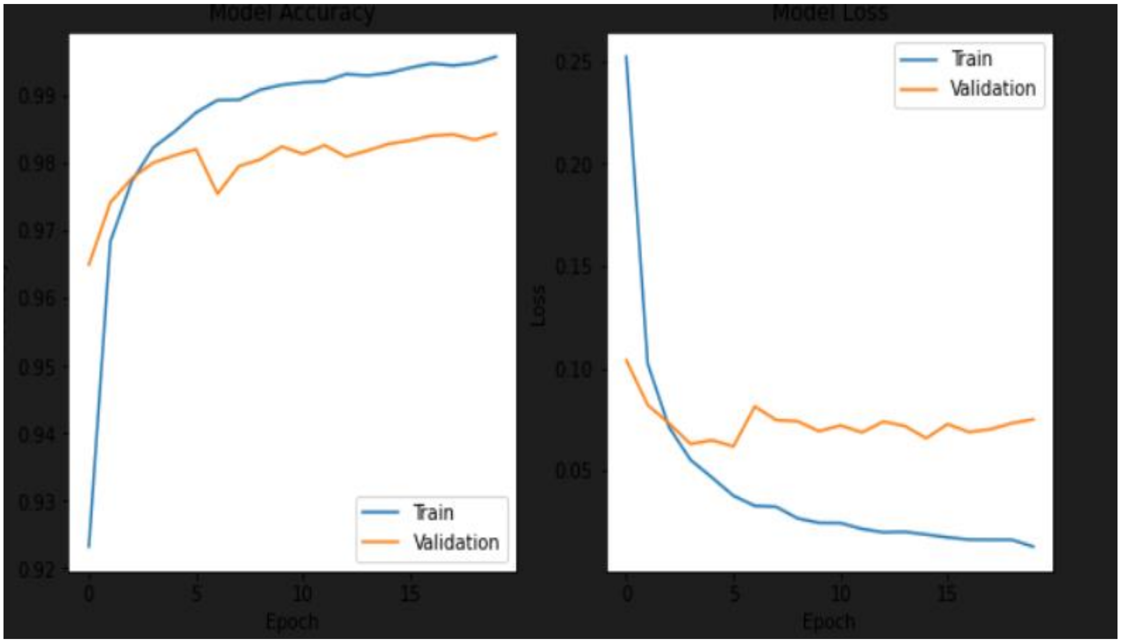


```
# plot the training and validation accuracy and loss curves
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='lower 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()
```

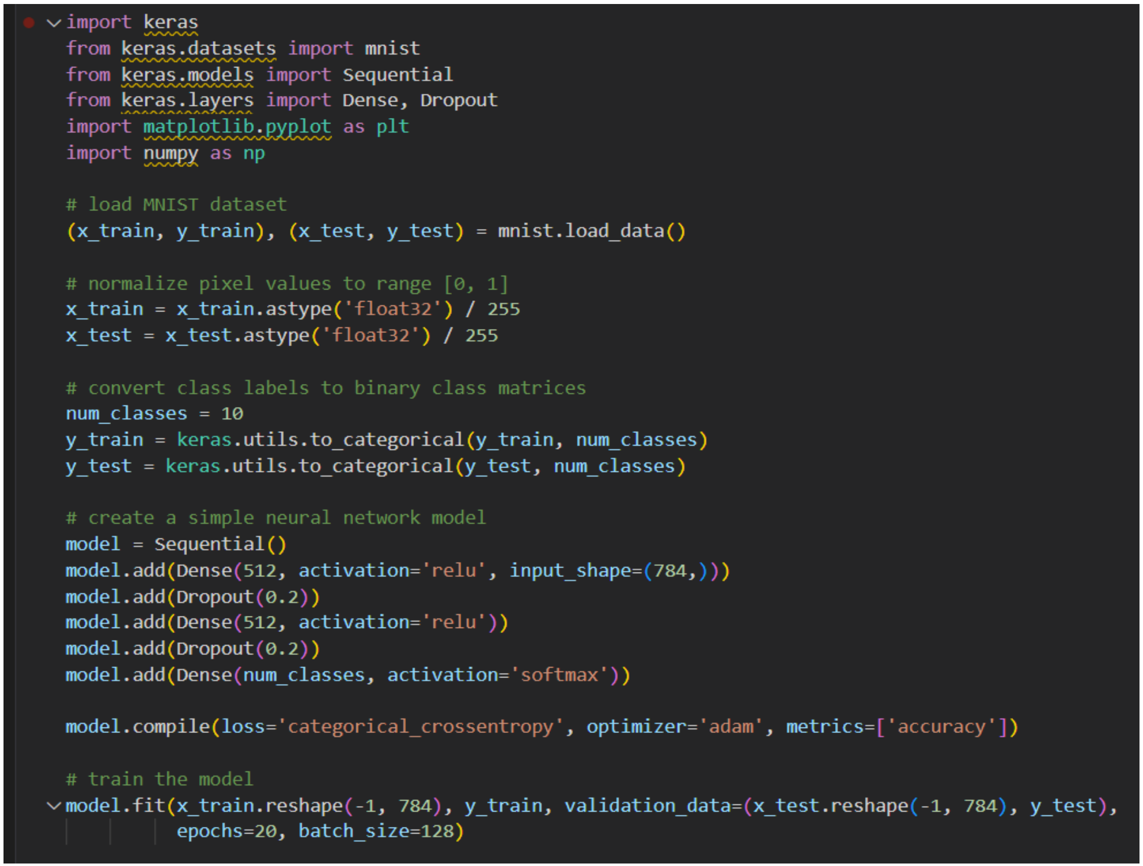
Output:



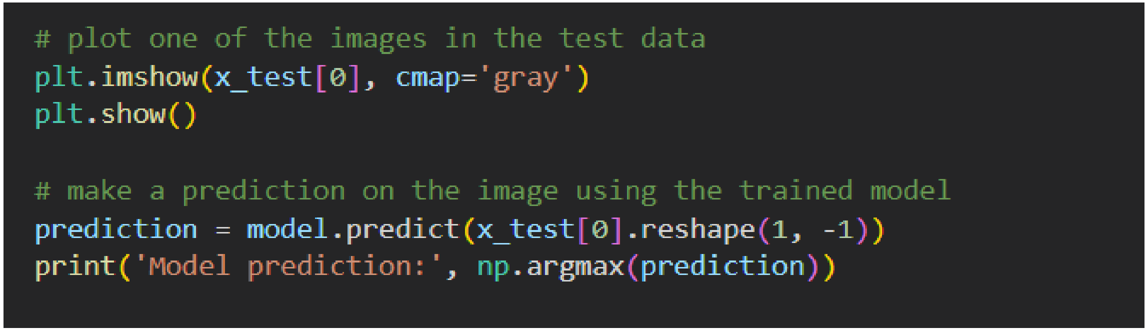
```
Epoch 1/20
469/469 [==============================] - 16s 27ms/step - loss: 0.2524 - accuracy: 0.9232 - val_loss: 0.1042 - val_accuracy: 0.9650
Epoch 2/20
469/469 [==============================] - 17s 36ms/step - loss: 0.1024 - accuracy: 0.9684 - val_loss: 0.0823 - val_accuracy: 0.9742
Epoch 3/20
469/469 [==============================] - 14s 29ms/step - loss: 0.0713 - accuracy: 0.9773 - val_loss: 0.0733 - val_accuracy: 0.9778
Epoch 4/20
469/469 [==============================] - 13s 28ms/step - loss: 0.0554 - accuracy: 0.9823 - val_loss: 0.0632 - val_accuracy: 0.9801
Epoch 5/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0468 - accuracy: 0.9847 - val_loss: 0.0651 - val_accuracy: 0.9812
Epoch 6/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0379 - accuracy: 0.9875 - val_loss: 0.0620 - val_accuracy: 0.9821
Epoch 7/20
469/469 [==============================] - 13s 28ms/step - loss: 0.0330 - accuracy: 0.9894 - val_loss: 0.0815 - val_accuracy: 0.9755
Epoch 8/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0325 - accuracy: 0.9894 - val_loss: 0.0749 - val_accuracy: 0.9796
Epoch 9/20
469/469 [==============================] - 15s 31ms/step - loss: 0.0269 - accuracy: 0.9909 - val_loss: 0.0743 - val_accuracy: 0.9806
Epoch 10/20
469/469 [==============================] - 12s 27ms/step - loss: 0.0247 - accuracy: 0.9916 - val_loss: 0.0694 - val_accuracy: 0.9825
Epoch 11/20
469/469 [==============================] - 14s 31ms/step - loss: 0.0246 - accuracy: 0.9920 - val_loss: 0.0723 - val_accuracy: 0.9814
Epoch 12/20
469/469 [==============================] - 12s 26ms/step - loss: 0.0217 - accuracy: 0.9922 - val_loss: 0.0688 - val_accuracy: 0.9827
Epoch 13/20
...
Epoch 19/20
469/469 [==============================] - 12s 26ms/step - loss: 0.0164 - accuracy: 0.9949 - val_loss: 0.0734 - val_accuracy: 0.9835
Epoch 20/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0131 - accuracy: 0.9958 - val_loss: 0.0752 - val_accuracy: 0.9844
```



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.

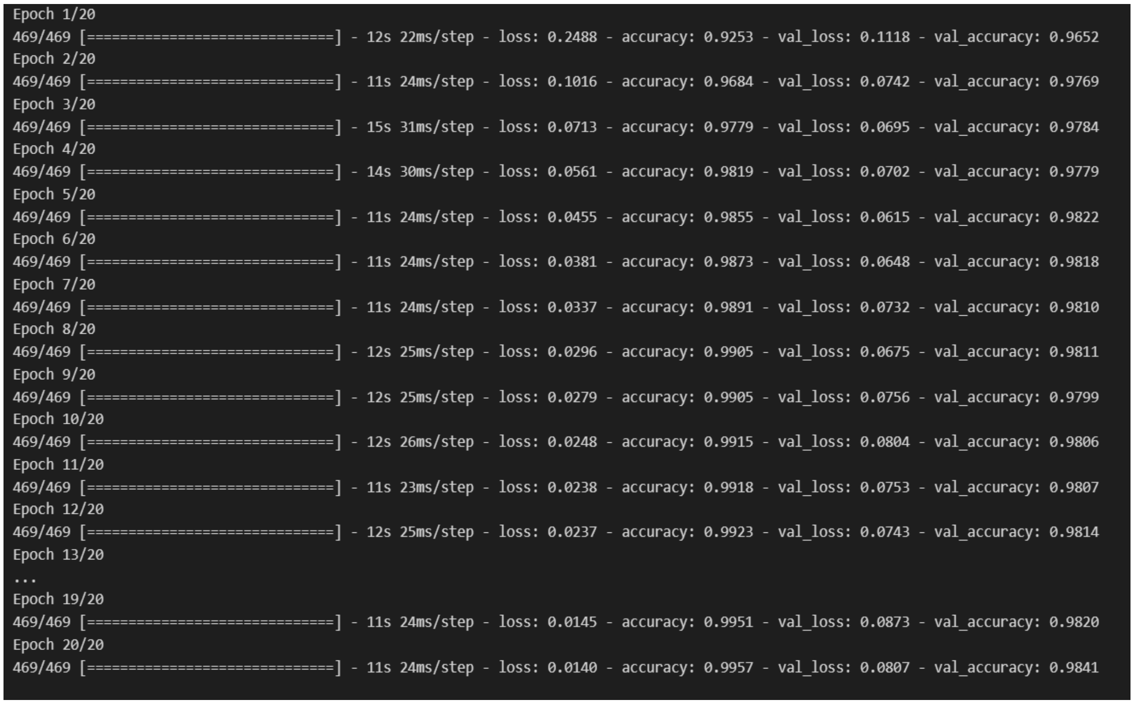


```
import keras
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout
import matplotlib.pyplot as plt
import numpy as np

# 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)
```

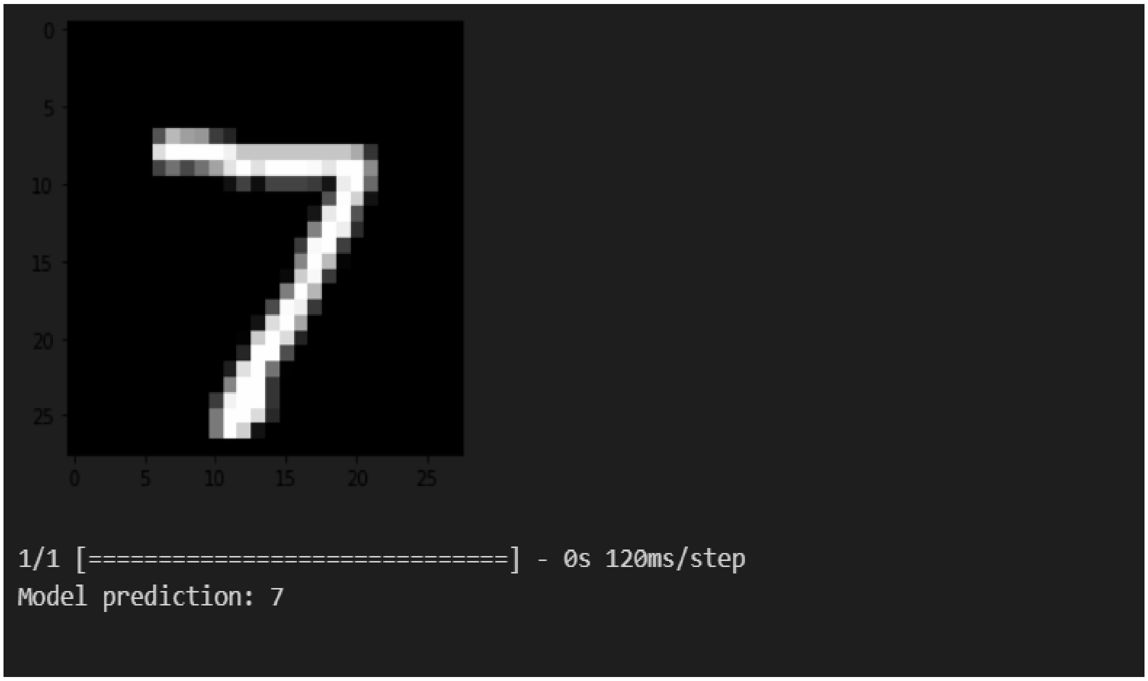


```
# plot one of the images in the test data
plt.imshow(x_test[0], cmap='gray')
plt.show()

# 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))
```

Output:

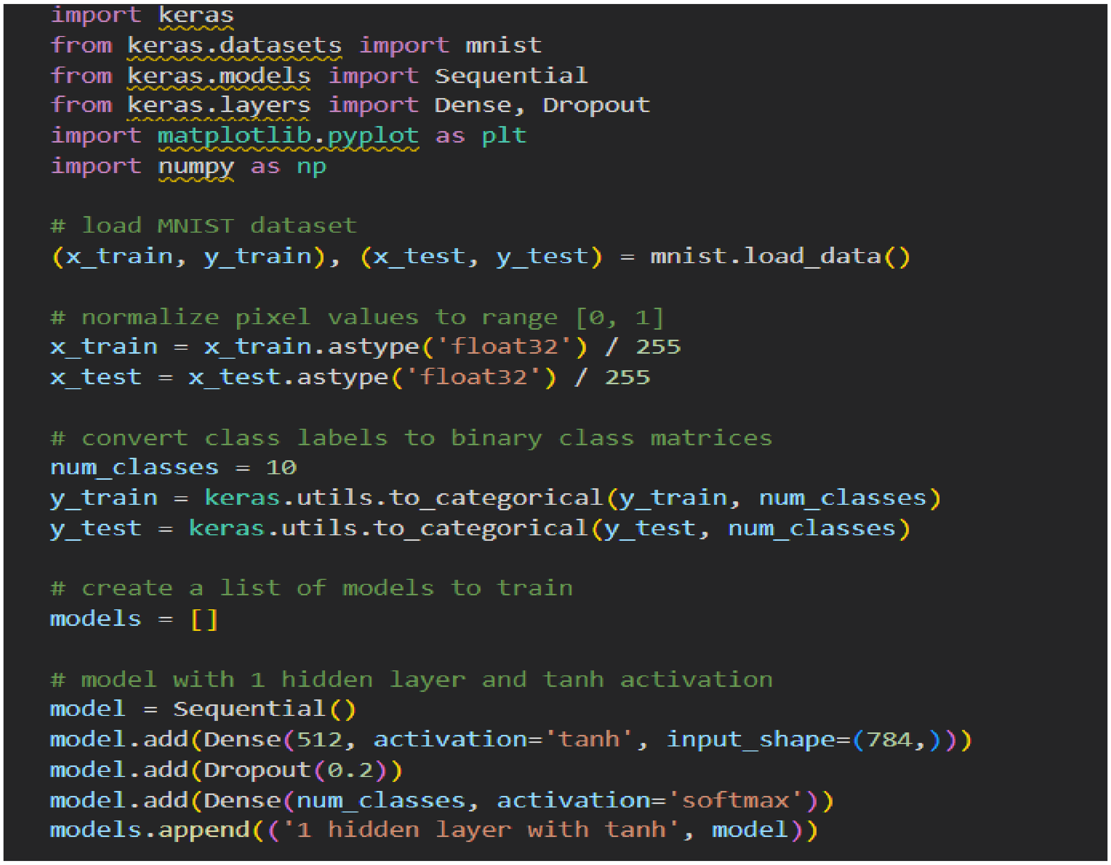


```
Epoch 1/20
469/469 [==============================] - 12s 22ms/step - loss: 0.2488 - accuracy: 0.9253 - val_loss: 0.1118 - val_accuracy: 0.9652
Epoch 2/20
469/469 [==============================] - 11s 24ms/step - loss: 0.1016 - accuracy: 0.9684 - val_loss: 0.0742 - val_accuracy: 0.9769
Epoch 3/20
469/469 [==============================] - 15s 31ms/step - loss: 0.0713 - accuracy: 0.9779 - val_loss: 0.0695 - val_accuracy: 0.9784
Epoch 4/20
469/469 [==============================] - 14s 30ms/step - loss: 0.0561 - accuracy: 0.9819 - val_loss: 0.0702 - val_accuracy: 0.9779
Epoch 5/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0455 - accuracy: 0.9855 - val_loss: 0.0615 - val_accuracy: 0.9822
Epoch 6/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0381 - accuracy: 0.9873 - val_loss: 0.0648 - val_accuracy: 0.9818
Epoch 7/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0337 - accuracy: 0.9891 - val_loss: 0.0732 - val_accuracy: 0.9810
Epoch 8/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0296 - accuracy: 0.9905 - val_loss: 0.0675 - val_accuracy: 0.9811
Epoch 9/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0279 - accuracy: 0.9905 - val_loss: 0.0756 - val_accuracy: 0.9799
Epoch 10/20
469/469 [==============================] - 12s 26ms/step - loss: 0.0248 - accuracy: 0.9915 - val_loss: 0.0804 - val_accuracy: 0.9806
Epoch 11/20
469/469 [==============================] - 11s 23ms/step - loss: 0.0238 - accuracy: 0.9918 - val_loss: 0.0753 - val_accuracy: 0.9807
Epoch 12/20
469/469 [==============================] - 12s 25ms/step - loss: 0.0237 - accuracy: 0.9923 - val_loss: 0.0743 - val_accuracy: 0.9814
Epoch 13/20
...
Epoch 19/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0145 - accuracy: 0.9951 - val_loss: 0.0873 - val_accuracy: 0.9820
Epoch 20/20
469/469 [==============================] - 11s 24ms/step - loss: 0.0140 - accuracy: 0.9957 - val_loss: 0.0807 - val_accuracy: 0.9841
```

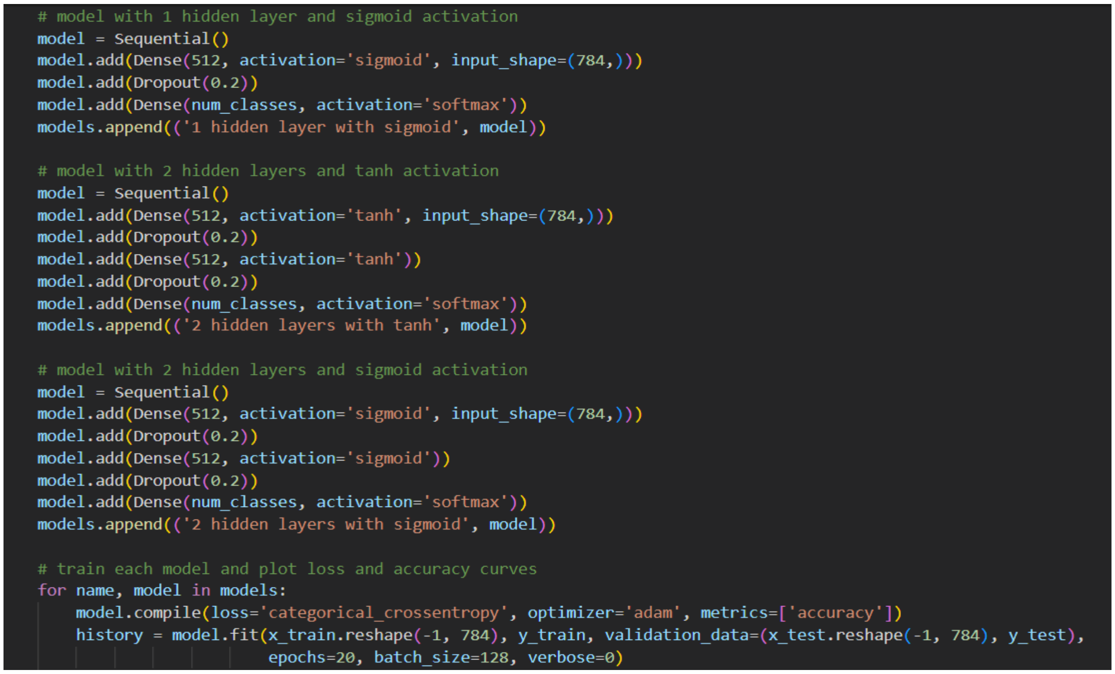


```
1/1 [==============================] - 0s 120ms/step
Model prediction: 7
```

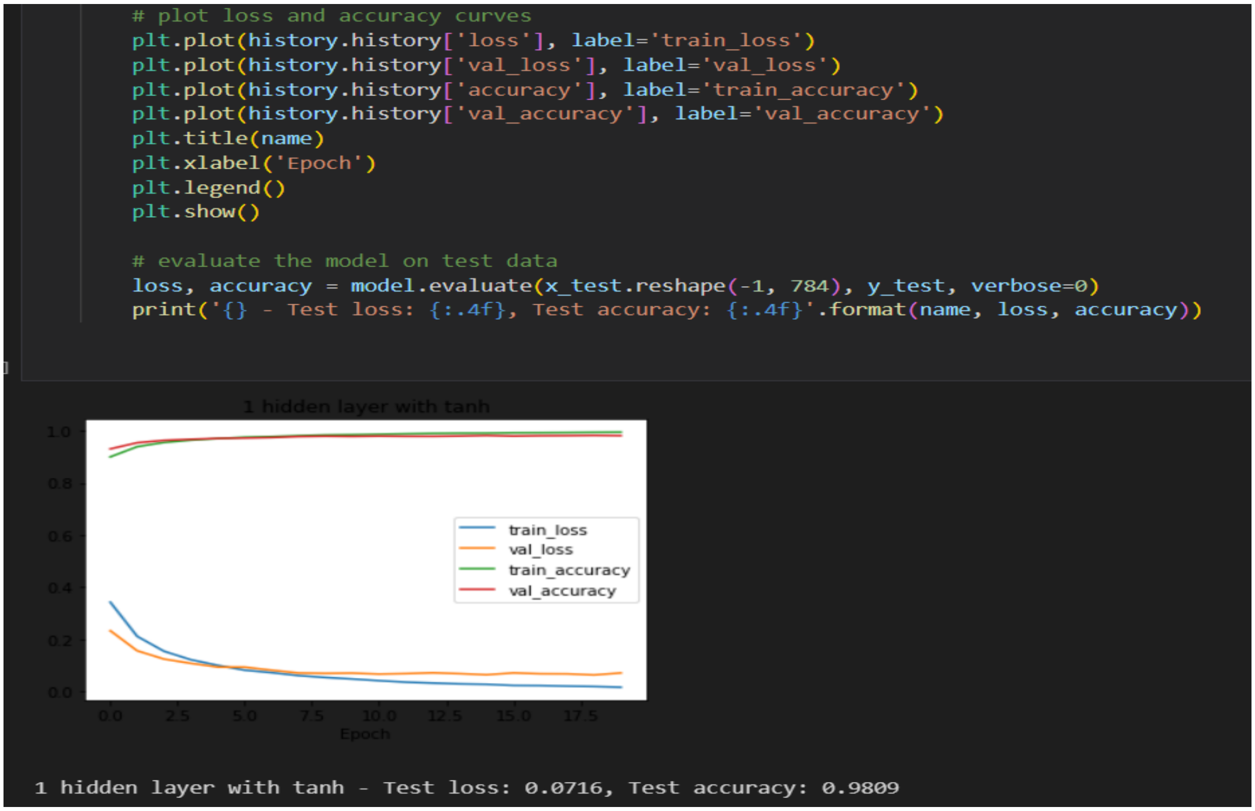
3. We had used 2 hidden layers and Relu activation. Try to change the number of hidden layer and the activation to tanh or sigmoid and see what happens.



```
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))
```

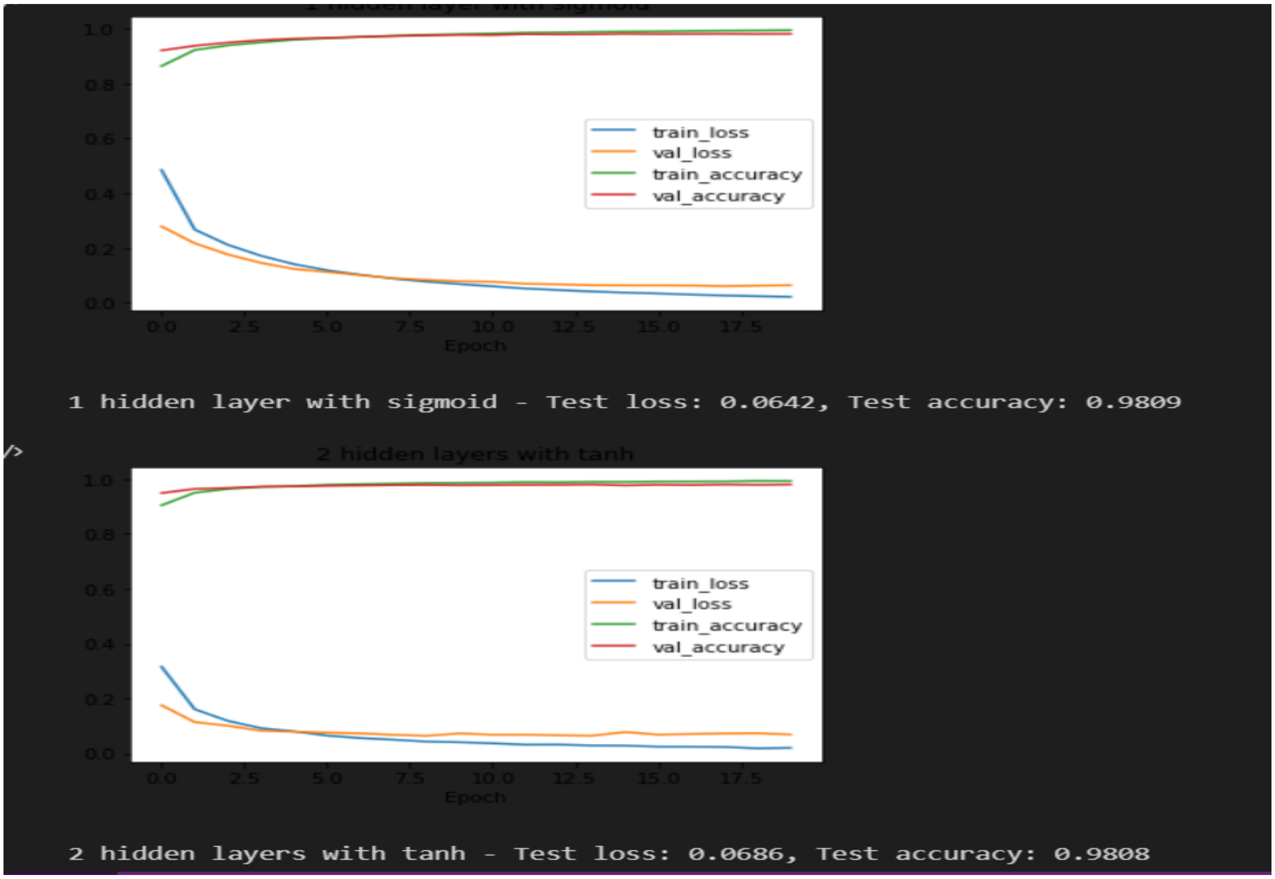


```
# model with 1 hidden layer and sigmoid activation
model = Sequential()
model.add(Dense(512, activation='sigmoid', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('1 hidden layer with sigmoid', model))

# 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)
```



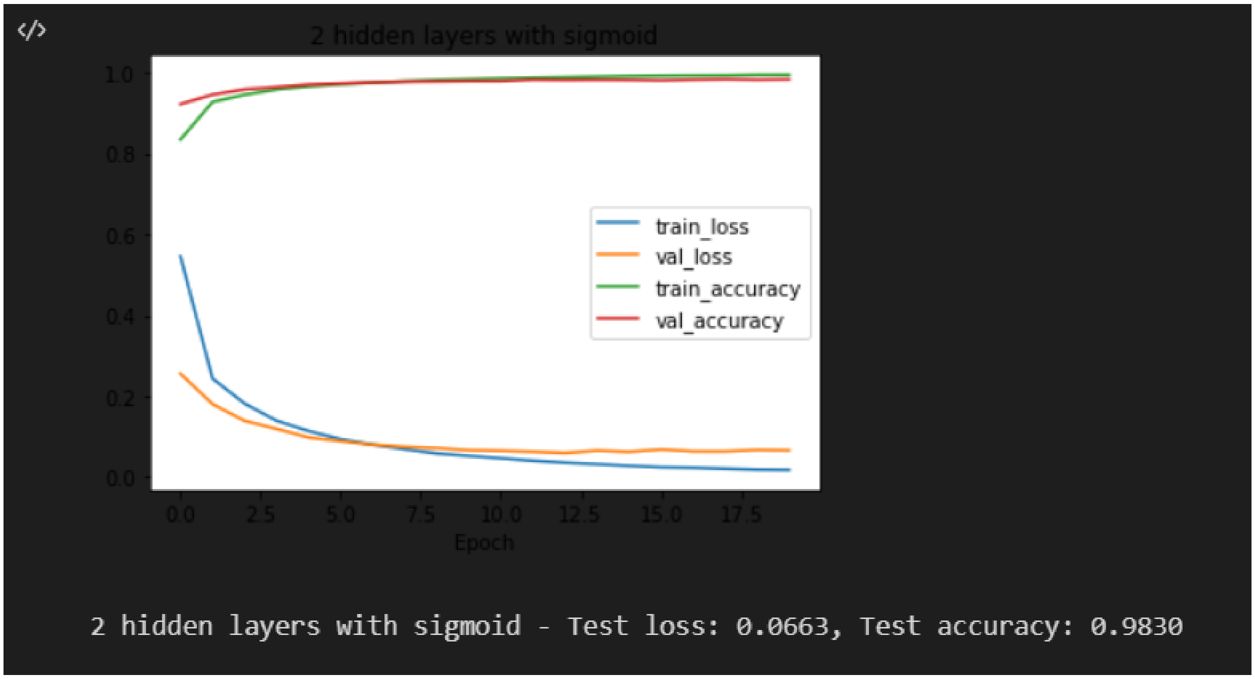
```
    # plot loss and accuracy curves
    plt.plot(history.history['loss'], label='train_loss')
    plt.plot(history.history['val_loss'], label='val_loss')
    plt.plot(history.history['accuracy'], label='train_accuracy')
    plt.plot(history.history['val_accuracy'], label='val_accuracy')
    plt.title(name)
    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: 0.0716, Test accuracy: 0.9809
```



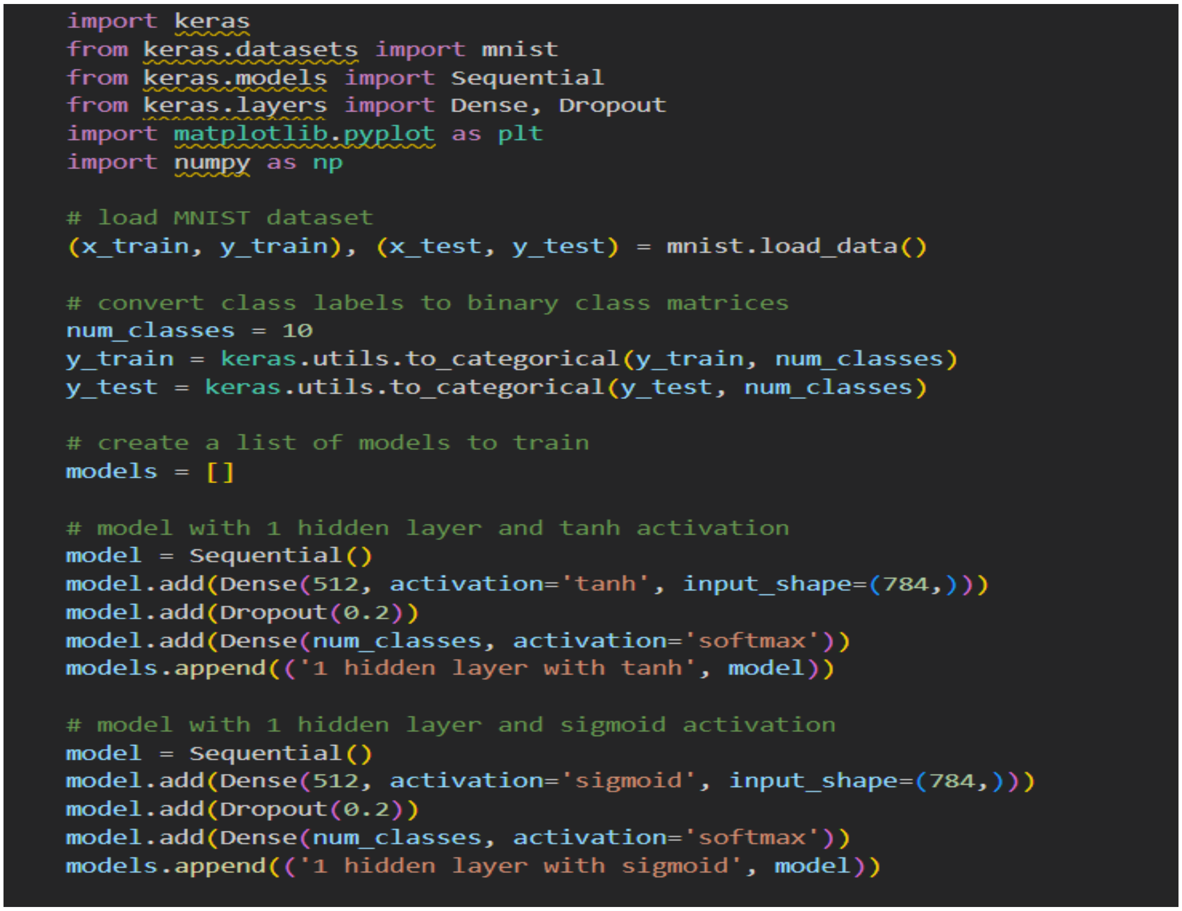
```
1 hidden layer with sigmoid - Test loss: 0.0642, Test accuracy: 0.9809
```

```
2 hidden layers with tanh - Test loss: 0.0686, Test accuracy: 0.9808
```

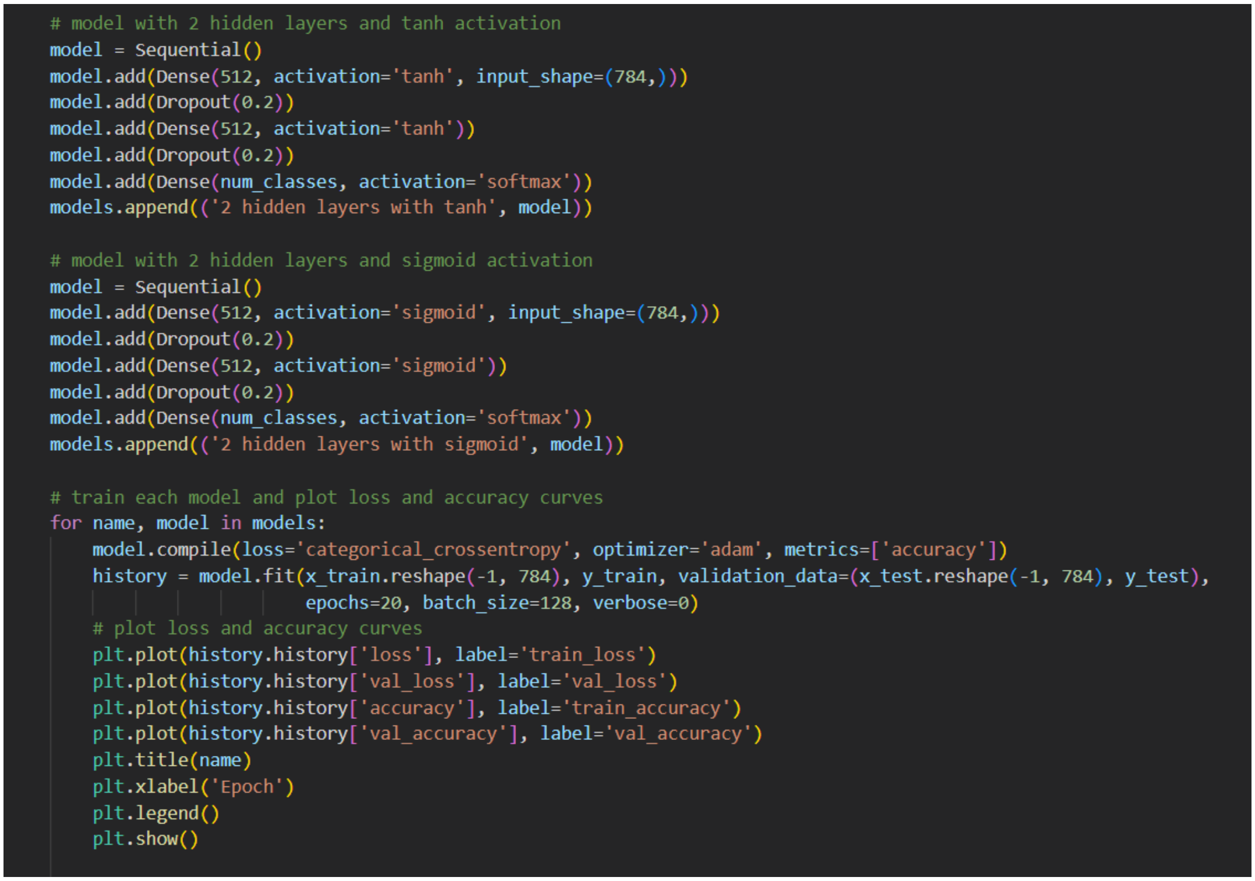


```
2 hidden layers with sigmoid - Test loss: 0.0663, Test accuracy: 0.9830
```

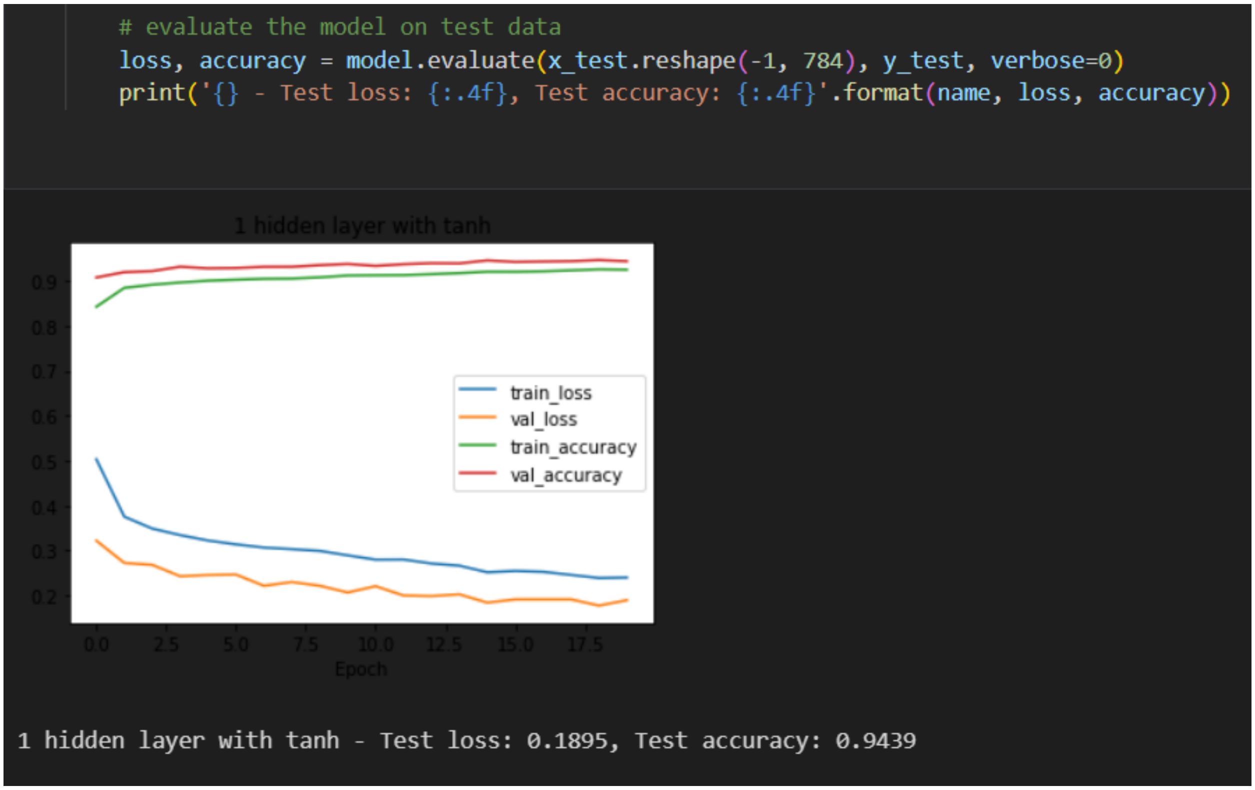
4. Run the same code without scaling the images and check the performance?



```
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))
```

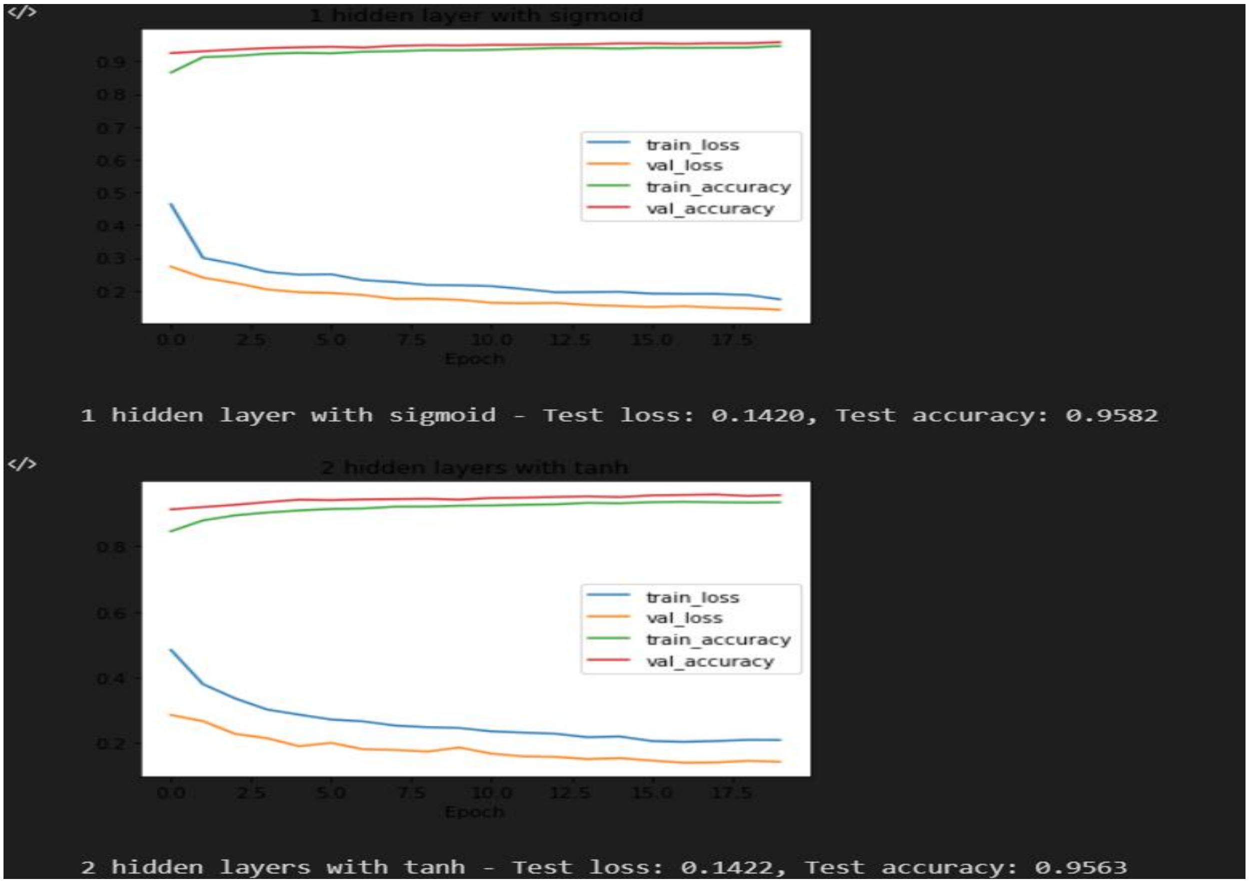


```
# model with 2 hidden layers and tanh activation
model = Sequential()
model.add(Dense(512, activation='tanh', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='tanh'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
models.append(('2 hidden 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()
```



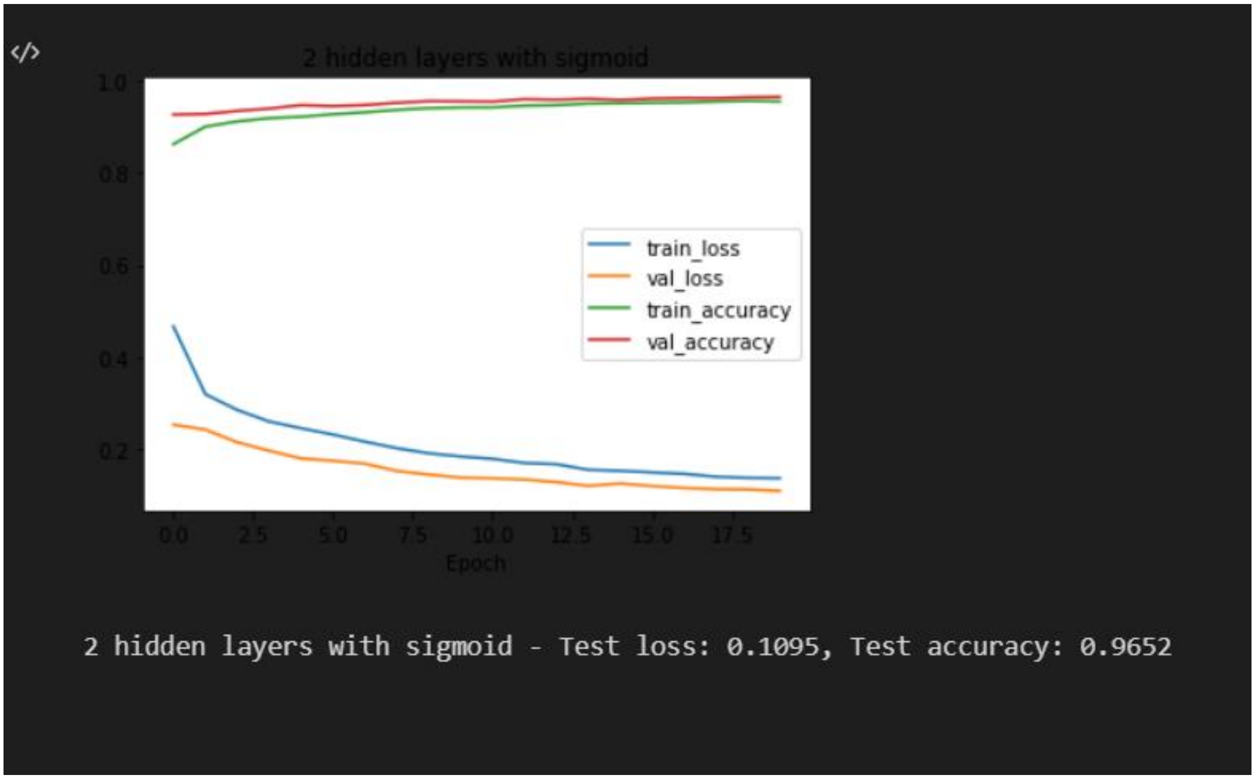
```
    # evaluate the model on test data
    loss, accuracy = model.evaluate(x_test.reshape(-1, 784), y_test, verbose=0)
    print('{} - Test loss: {:.4f}, Test accuracy: {:.4f}'.format(name, loss, accuracy))
```

```
1 hidden layer with tanh - Test loss: 0.1895, Test accuracy: 0.9439
```



```
1 hidden layer with sigmoid - Test loss: 0.1420, Test accuracy: 0.9582
```

```
2 hidden layers with tanh - Test loss: 0.1422, Test accuracy: 0.9563
```



```
2 hidden layers with sigmoid - Test loss: 0.1095, Test accuracy: 0.9652
```