```
import numpy as np
import pandas as pd
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.layers import Conv2D
from keras.optimizers import Adam
from keras.utils import np_utils
```

```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist
11490434/11490434 [==============================] - 0s 0us/step
```

```
print(x_train.shape)
```

```
(60000, 28, 28)
```

```
print(x_test.shape)
```

```
(10000, 28, 28)
```

```
x_train[0]
```

```
        253, 198, 182, 247, 241,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  80, 156, 107, 253, 253,
        205,  11,   0,  43, 154,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,  14,   1, 154, 253,
         90,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 139, 253,
        190,   2,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  11, 190,
        253,  70,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  35,
        241, 225, 160, 108,   1,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         81, 240, 253, 253, 119,  25,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,  45, 186, 253, 253, 150,  27,   0,   0,   0,   0,   0,   0,
          0,   0]
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,  16,  93, 252, 253, 187,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0, 249, 253, 249,  64,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,  46, 130, 183, 253, 253, 207,   2,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  39,
        148, 229, 253, 253, 253, 250, 182,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  24, 114, 221,
        253, 253, 253, 253, 201,  78,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  23,  66, 213, 253, 253,
        253, 253, 198,  81,   2,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  18, 171, 219, 253, 253, 253, 253,
        195,  80,   9,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,  55, 172, 226, 253, 253, 253, 253, 244, 133,
         11,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0, 136, 253, 253, 253, 212, 135, 132,  16,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
```

```
import matplotlib.pyplot as plt
```

```
plt.imshow(x_train[2])
```

```
<matplotlib.image.AxesImage at 0x7f0b7a031550>
```

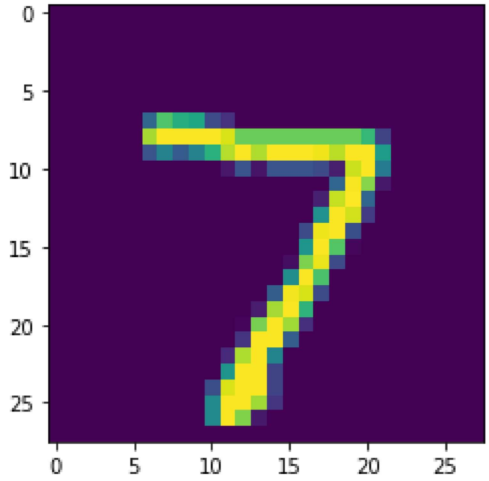
```
x_test[0]
```

```
         0,   0],
       [  0,   0,   0,   0,   0,   0,  67, 114,  72, 114, 163, 227, 254,
        225, 254, 254, 254, 250, 229, 254, 254, 140,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  17,  66,
         14,  67,  67,  67,  59,  21, 236, 254, 106,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,  83, 253, 209,  18,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,  22, 233, 255,  83,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0, 129, 254, 238,  44,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,  59, 249, 254,  62,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0, 133, 254, 187,   5,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   9, 205, 248,  58,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0, 126, 254, 182,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,  75, 251, 240,  57,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         19, 221, 254, 166,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
        203, 254, 219,  35,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  38,
        254, 254,  77,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  31, 224,
        254, 115,   1,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 133, 254,
        254,  52,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  61, 242, 254,
        254,  52,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0, 121, 254, 254,
        219,  40,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0]
```

```
                0,    0],
             [  0,    0,    0,    0,    0,    0,    0,    0,    0,    0, 121, 254, 207,
               18,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
                0,    0],
             [  0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
                0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
                0,    0]], dtype=uint8)
```

```
plt.imshow(x_test[0])
```

```
<matplotlib.image.AxesImage at 0x7f0b79b1c090>
```



```
x_train = x_train.reshape(60000,28,28,1).astype('float32')
x_test = x_test.reshape(10000,28,28,1).astype('float32')
```

```
number_of_classes = 10
y_train = np_utils.to_categorical(y_train,number_of_classes)
y_test = np_utils.to_categorical(y_test,number_of_classes)
```

```
y_train[0]
```

```
array([0., 0., 0., 0., 0., 1., 0., 0., 0., 0.], dtype=float32)
```

```
model = Sequential()
model.add(Conv2D(64,(3,3),input_shape = (28,28,1), activation='relu'))
model.add(Conv2D(32,(3,3), activation = 'relu'))
model.add(Flatten())
model.add(Dense(number_of_classes,activation = 'softmax'))
```

```
model.compile(loss='categorical_crossentropy', optimizer='Adam', metrics=['accuracy'])
```

```
model.fit(x_train,y_train, validation_data = (x_test,y_test),epochs=5,batch_size=32)
```

```
Epoch 1/5
```

```
1875/1875 [==============================] - 16s 4ms/step - loss: 0.2258 - accuracy: 0.
Epoch 2/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0647 - accuracy: 0.9
Epoch 3/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0452 - accuracy: 0.9
Epoch 4/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0362 - accuracy: 0.9
Epoch 5/5
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0292 - accuracy: 0.9
<keras.callbacks.History at 0x7f0b7007b150>
```

```
metrics= model.evaluate(x_test,y_test,verbose=0)
print(metrics)
```

```
[0.09644468873739243, 0.9768999814987183]
```

```
prediction = model.predict(x_test[:4])
print(prediction)
```

```
1/1 [==============================] - 0s 186ms/step
[[3.99105899e-14 2.34958310e-20 3.46917738e-11 2.50990700e-12
  1.71917132e-17 7.51082643e-16 1.66998525e-20 1.00000000e+00
  1.77415591e-14 1.59968809e-11]
 [3.07460585e-11 1.29253986e-11 1.00000000e+00 4.23759301e-17
  8.87018693e-18 5.91285370e-18 5.10872677e-09 1.60456398e-20
  1.33883920e-12 2.26489302e-23]
 [5.32193347e-08 9.99408960e-01 3.10939390e-06 5.19040200e-10
  2.49341392e-05 5.07483310e-05 9.09071673e-07 4.54493332e-08
  5.11370657e-04 1.12538923e-09]
 [1.00000000e+00 2.02566553e-15 2.88732898e-14 1.28468267e-14
  4.11677097e-16 1.22025003e-13 5.62831448e-11 1.07858773e-16
  3.33526401e-11 4.15842860e-10]]
```

```
model.save('models/mnistCNN.h5')
```

```
from tensorflow.keras.models import load_model
model = load_model("/content/models/mnistCNN.h5")
```

```
from PIL import Image
import numpy as np
for index in range(4):
  img = Image.open('/content/sample_data/data/1.png').convert('L')
  img = img.resize((28,28))
  im2arr=np.array(img)
  im2arr = im2arr.reshape(1,28,28,1)
  y_pred = model.predict(im2arr)
  print(y_pred)
```

```
1/1 [==============================] - 0s 15ms/step
```

```
[[0.09914009 0.10481779 0.09998827 0.09787922 0.09672169 0.09803257
  0.09807642 0.09996875 0.1058826  0.09949266]]
1/1 [==============================] - 0s 17ms/step
[[0.09914009 0.10481779 0.09998827 0.09787922 0.09672169 0.09803257
  0.09807642 0.09996875 0.1058826  0.09949266]]
1/1 [==============================] - 0s 16ms/step
[[0.09914009 0.10481779 0.09998827 0.09787922 0.09672169 0.09803257
  0.09807642 0.09996875 0.1058826  0.09949266]]
1/1 [==============================] - 0s 16ms/step
[[0.09914009 0.10481779 0.09998827 0.09787922 0.09672169 0.09803257
  0.09807642 0.09996875 0.1058826  0.09949266]]
```

✓ 0s completed at 1:46 PM ● ✕