```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras import models, datasets, layers
import matplotlib.pyplot as plt
import matplotlib.image as mp
```

```
(train_images,train_labels),(test_images,test_labels)=datasets.mnist.load_data()
```

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

```
print('x_tain: ', train_images.shape)
print('y_tain: ', train_labels.shape)
print('x_test: ', test_images.shape)
print('y_test: ', test_labels.shape)
```

```
x_tain:  (60000, 28, 28)
y_tain:  (60000,)
x_test:  (10000, 28, 28)
y_test:  (10000,)
```

```
pd.DataFrame(train_images[0])
```

	0	1	2	3	4	5	6	7	8	9	...	18	19	20	21	22	23	24	25	26	27
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	...	175	26	166	255	247	127	0	0	0	0
6	0	0	0	0	0	0	0	0	30	36	...	225	172	253	242	195	64	0	0	0	0
7	0	0	0	0	0	0	0	49	238	253	...	93	82	82	56	39	0	0	0	0	0
8	0	0	0	0	0	0	0	18	219	253	...	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	80	156	...	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	14	...	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	...	25	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	...	150	27	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	...	253	187	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	...	253	249	64	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	...	253	207	2	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	...	250	182	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	...	78	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	23	66	...	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	18	171	219	253	...	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	55	172	226	253	253	253	...	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	136	253	253	253	212	135	...	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

28 rows × 28 columns

```
train_images=train_images/255
test_images=test_images/255
```

```
model=models.Sequential()
model.add(layers.Flatten(input_shape=(28,28,1)))
model.add(layers.Dense(32,activation='relu'))
model.add(layers.Dense(16,activation='relu'))
model.add(layers.Dense(10,activation='softmax'))
```

```
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 32)                25120

 dense_1 (Dense)             (None, 16)                528

 dense_2 (Dense)             (None, 10)                170

=================================================================
Total params: 25818 (100.85 KB)
Trainable params: 25818 (100.85 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
h = model.fit(train_images,train_labels, epochs=10, validation_data = (test_images,test_labels))
```

```
Epoch 1/10
1875/1875 [==============================] - 13s 4ms/step - loss: 0.4021 - accuracy: 0.8810 - val_loss: 0.2010 - val_accuracy: 0.944
Epoch 2/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1852 - accuracy: 0.9459 - val_loss: 0.1558 - val_accuracy: 0.9566
Epoch 3/10
1875/1875 [==============================] - 7s 3ms/step - loss: 0.1459 - accuracy: 0.9569 - val_loss: 0.1463 - val_accuracy: 0.9591
Epoch 4/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1245 - accuracy: 0.9632 - val_loss: 0.1365 - val_accuracy: 0.9617
Epoch 5/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.1102 - accuracy: 0.9675 - val_loss: 0.1336 - val_accuracy: 0.9614
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0989 - accuracy: 0.9704 - val_loss: 0.1166 - val_accuracy: 0.9661
Epoch 7/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0892 - accuracy: 0.9735 - val_loss: 0.1166 - val_accuracy: 0.9676
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.0809 - accuracy: 0.9758 - val_loss: 0.1137 - val_accuracy: 0.9689
Epoch 9/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.0751 - accuracy: 0.9769 - val_loss: 0.1240 - val_accuracy: 0.9687
Epoch 10/10
1875/1875 [==============================] - 7s 3ms/step - loss: 0.0705 - accuracy: 0.9785 - val_loss: 0.1128 - val_accuracy: 0.9701
```

```
score = model.evaluate(test_images,test_labels)
print("test loss :", score[0])
print("test accuracy :", score[1])
```

```
313/313 [==============================] - 1s 3ms/step - loss: 0.1128 - accuracy: 0.9701
test loss : 0.11276522278785706
test accuracy : 0.9700999855995178
```

```
model_name="file.h5"
model.save(model_name,save_format='h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3000: UserWarning: You are saving your model as an HDF5 file vi
  saving_api.save_model(
```

```
loaded_model = tf.keras.models.load_model(model_name)
```

```
predictions_one_hot = loaded_model.predict([test_images])
```

```
313/313 [==============================] - 1s 2ms/step
```

```
print("predications one hot :", predictions_one_hot.shape)
```

```
predications one hot : (10000, 10)
```

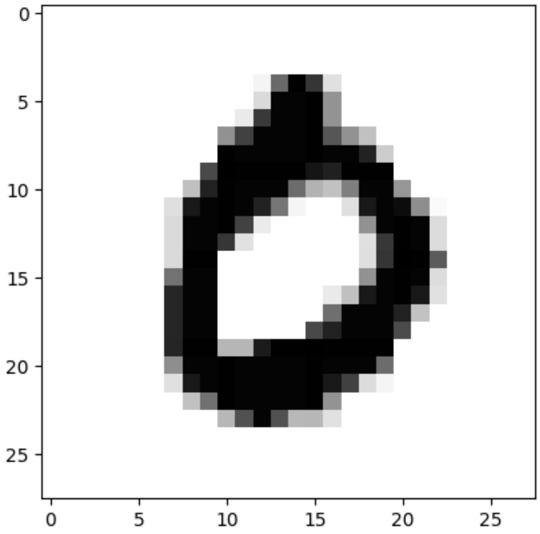
```
predictions=np.argmax(predictions_one_hot, axis=1)
pd.DataFrame(predictions)
```

```
         0
   0     7
   1     2
   2     1
   3     0
   4     4
  ...   ...
9995     2
9996     3
9997     4
9998     5
9999     6
10000 rows × 1 columns
```

```
print(predictions[3])
```

```
0
```

```
plt.imshow(test_images[3].reshape((28,28)),cmap=plt.cm.binary)
plt.show()
```


0
5
10
15
20
25
0
5
10
15
20
25