

```
import tensorflow as tf
from keras.models import Sequential
from keras.datasets import mnist
import matplotlib.pyplot as plt
import numpy as np
import random
```

```
(x_train,y_train),(x_test,y_test)=mnist.load_data()
x_train=x_train/255
x_test=x_test/255
```

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

```
import keras
model=keras.Sequential()
model.add(keras.layers.Flatten(input_shape=(28,28)))
model.add(keras.layers.Dense(128,activation='relu'))
model.add(keras.layers.Dense(10,activation='softmax'))
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 784)	0
dense (Dense)	(None, 128)	100480
dense_1 (Dense)	(None, 10)	1290
Total params: 101770 (397.54 KB)		
Trainable params: 101770 (397.54 KB)		
Non-trainable params: 0 (0.00 Byte)		

```
model.compile(optimizer='sgd', loss='sparse_categorical_crossentropy', metrics=["Accuracy"])
```

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

```
Epoch 1/5
1875/1875 [=====] - 12s 6ms/step - loss: 0.6471 - Accuracy: 0.8358 - val_loss: 0.3569 - val_Accuracy: 0.902
Epoch 2/5
1875/1875 [=====] - 14s 7ms/step - loss: 0.3390 - Accuracy: 0.9053 - val_loss: 0.2948 - val_Accuracy: 0.915
Epoch 3/5
1875/1875 [=====] - 14s 8ms/step - loss: 0.2911 - Accuracy: 0.9184 - val_loss: 0.2628 - val_Accuracy: 0.927
Epoch 4/5
1875/1875 [=====] - 6s 3ms/step - loss: 0.2611 - Accuracy: 0.9273 - val_loss: 0.2407 - val_Accuracy: 0.9315
Epoch 5/5
1875/1875 [=====] - 7s 4ms/step - loss: 0.2380 - Accuracy: 0.9336 - val_loss: 0.2229 - val_Accuracy: 0.9365
```

```
test_loss,test_acc=model.evaluate(x_test,y_test)
```

```
313/313 [=====] - 1s 2ms/step - loss: 0.2229 - Accuracy: 0.9369
```

```
print("Loss=%.3f"%test_loss)
print("Accuracy=%.3f"%test_acc)
```

```
Loss=0.223
Accuracy=0.937
```

```
n=random.randint(0,999)
plt.imshow(x_test[n])
plt.show()
```



```
prediction=model.predict(x_test)
print("The handwritten number in the image is %d"%np.argmax(prediction[n]))
```

```
313/313 [=====] - 1s 2ms/step
The handwritten number in the image is 4
```

