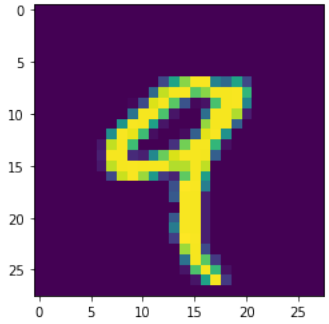
```
#importing the necessary modules
from keras.datasets import mnist
import numpy as np
import matplotlib.pyplot as plt
```

```
#loading data
(xtrain,ytrain),(xtest,ytest)=mnist.load_data()
```

```
#What mnist images look like
import matplotlib.pyplot as plt
print("Training data:")
plt.imshow(xtrain[4])
plt.show()
print("Label of this image is",ytrain[4])
```

```
Training data:
```



```
Label of this image is 9
```

```
#reshaping data as needed by the model
xtrain=np.reshape(xtrain,(-1,28,28,1))
xtest=np.reshape(xtest,(-1,28,28,1))
xtrain.shape,xtest.shape,ytrain.shape,ytest.shape
```

```
((60000, 28, 28, 1), (10000, 28, 28, 1), (60000,), (10000,))
```

```
#normalising
xtrain=xtrain/255
xtest=xtest/255
```

```
#implementing one hot encoding
from keras.utils.np_utils import to_categorical
y_train = to_categorical(ytrain, num_classes=10)
y_test = to_categorical(ytest, num_classes=10)
```

```
#importing the model
from keras.models import Sequential
```

```
#creating model object
model=Sequential()
```

```
#importing layers
from keras.layers import Conv2D,MaxPooling2D,Flatten,Dense,Dropout
```

```
#adding layers and forming the model
model.add(Conv2D(32,kernel_size=5,strides=1,padding="Same",activation="relu",input_shape=(28,28,1)))
model.add(MaxPooling2D(padding="same"))

model.add(Conv2D(64,kernel_size=5,strides=1,padding="same",activation="relu"))
model.add(MaxPooling2D(padding="same"))

model.add(Flatten())

model.add(Dense(1024,activation="relu"))
model.add(Dropout(0.2))
model.add(Dense(10,activation="sigmoid"))
```

```
#compiling
model.compile(optimizer="adam",loss="categorical_crossentropy",metrics=["accuracy"])
```

```
#training the model
model.fit(xtrain,y_train,batch_size=100,epochs=5,validation_data=(xtest,y_test))
```

```
Epoch 1/5
600/600 [==============================] - 173s 287ms/step - loss: 0.1317 - accuracy: 0.9599 - val_loss: 0.0449 - val_accuracy: 0.98
Epoch 2/5
600/600 [==============================] - 172s 286ms/step - loss: 0.0387 - accuracy: 0.9879 - val_loss: 0.0290 - val_accuracy: 0.98
Epoch 3/5
600/600 [==============================] - 171s 285ms/step - loss: 0.0264 - accuracy: 0.9917 - val_loss: 0.0285 - val_accuracy: 0.99
Epoch 4/5
600/600 [==============================] - 170s 284ms/step - loss: 0.0195 - accuracy: 0.9939 - val_loss: 0.0224 - val_accuracy: 0.99
Epoch 5/5
600/600 [==============================] - 169s 282ms/step - loss: 0.0149 - accuracy: 0.9954 - val_loss: 0.0244 - val_accuracy: 0.99
<keras.callbacks.History at 0x7f0607e95490>
```

```
#model train and test scores
model.evaluate(xtrain,y_train),model.evaluate(xtest,y_test)
```

```
1875/1875 [==============================] - 45s 24ms/step - loss: 0.0088 - accuracy: 0.9977
313/313 [==============================] - 8s 24ms/step - loss: 0.0244 - accuracy: 0.9919
([0.008804108947515488, 0.9976500272750854],
 [0.024355504661798477, 0.9919000267982483])
```