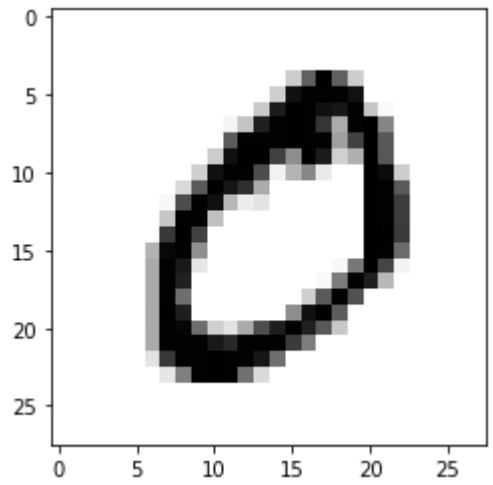
```
from keras.datasets import mnist
from keras import models
from keras import layers
from keras.utils import to_categorical
```

```
Using TensorFlow backend.
```

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

```
Downloading data from https://s3.amazonaws.com/img-datasets/mnist.npz
11493376/11490434 [==============================] - 1s 0us/step
```

```
digit = train_images[1]
import matplotlib.pyplot as plt
plt.imshow(digit, cmap=plt.cm.binary)
plt.show()
```




```
train_images.shape
```

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

```
train_images.dtype
```

```
dtype('uint8')
```

```
print(train_images)
```

```
      [[[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]
        ...
```

```
test_labels
```

```
array([7, 2, 1, ..., 4, 5, 6], dtype=uint8)
```

The network architecture

```
network = models.Sequential()
network.add(layers.Dense(512, activation='relu', input_shape=(28 * 28,)))
network.add(layers.Dense(10, activation='softmax'))
```

```
network.compile(optimizer='rmsprop',
loss='categorical_crossentropy',
metrics=['accuracy'])
```

```
train_images = train_images.reshape((60000, 28 * 28))
train_images = train_images.astype('float32') / 255
test_images = test_images.reshape((10000, 28 * 28))
test_images = test_images.astype('float32') / 255
```

```
...
```

```
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
```

```
network.fit(train_images, train_labels, epochs=5, batch_size=128)
```

```
network.fit(train_images, train_labels, epochs=5, batch_size=128)
```

```
Epoch 1/5
60000/60000 [==============================] - 4s 62us/step - loss: 0.2569 - accuracy: 0.9251
Epoch 2/5
60000/60000 [==============================] - 4s 59us/step - loss: 0.1033 - accuracy: 0.9692
Epoch 3/5
60000/60000 [==============================] - 3s 57us/step - loss: 0.0687 - accuracy: 0.9793
Epoch 4/5
60000/60000 [==============================] - 3s 56us/step - loss: 0.0505 - accuracy: 0.9848
Epoch 5/5
60000/60000 [==============================] - 4s 58us/step - loss: 0.0370 - accuracy: 0.9890
<keras.callbacks.callbacks.History at 0x7fc8023d4d68>
```

```
test_loss, test_acc = network.evaluate(test_images, test_labels)
```

```
10000/10000 [==============================] - 0s 45us/step
```

```
print('test_acc:', test_acc)
```

```
test_acc: 0.9805999994277954
```