In [1]:

```
import tensorflow as tf
from tensorflow import keras
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import random
%matplotlib inline
```

In [2]:

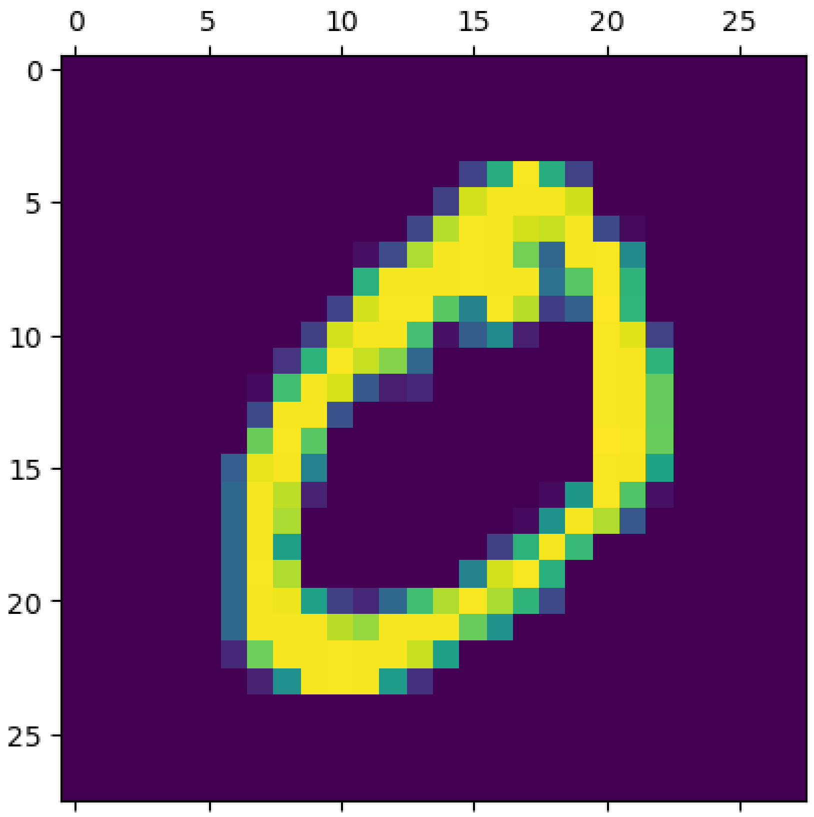
```
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

In [3]:

```
plt.matshow(x_train[1])
```

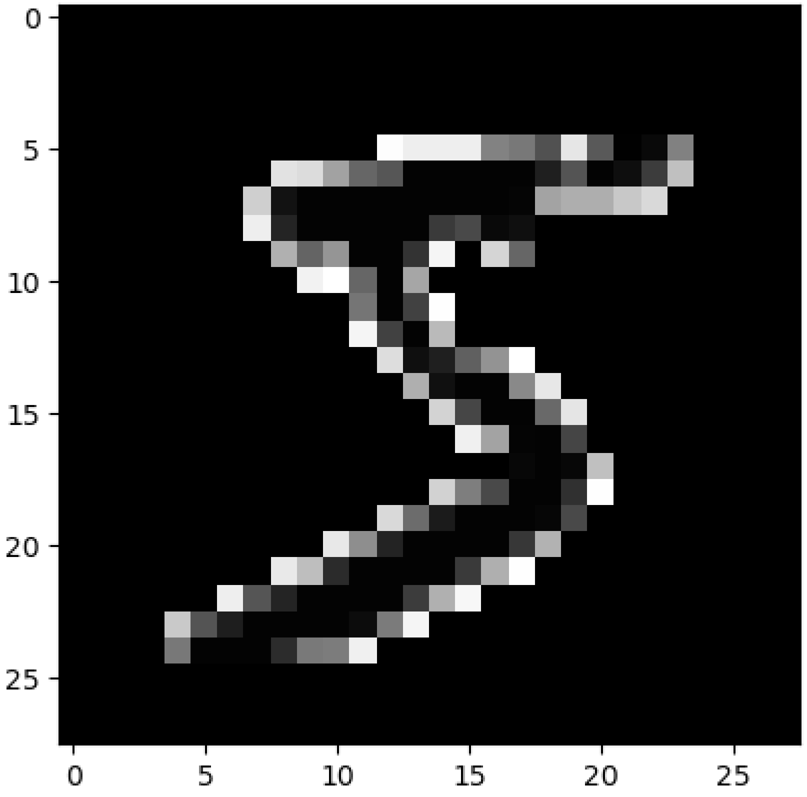
Out[3]:

```
<matplotlib.image.AxesImage at 0x15634a72200>
```



In [4]:
```
plt.imshow(-x_train[0], cmap="gray")
```

Out[4]:
```
<matplotlib.image.AxesImage at 0x1563865e5c0>
```



In [5]:
```
x_train = x_train / 255
x_test = x_test / 255
```

In [6]:
```
model = keras.Sequential([
keras.layers.Flatten(input_shape=(28, 28)),
keras.layers.Dense(128, activation="relu"),
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)
_________________________________________________________________
```

In [7]:
```
model.compile(optimizer="sgd",
loss="sparse_categorical_crossentropy",
metrics=['accuracy'])
```

In [8]:
```
history=model.fit(x_train,
y_train,validation_data=(x_test,y_test),epochs=10)
```

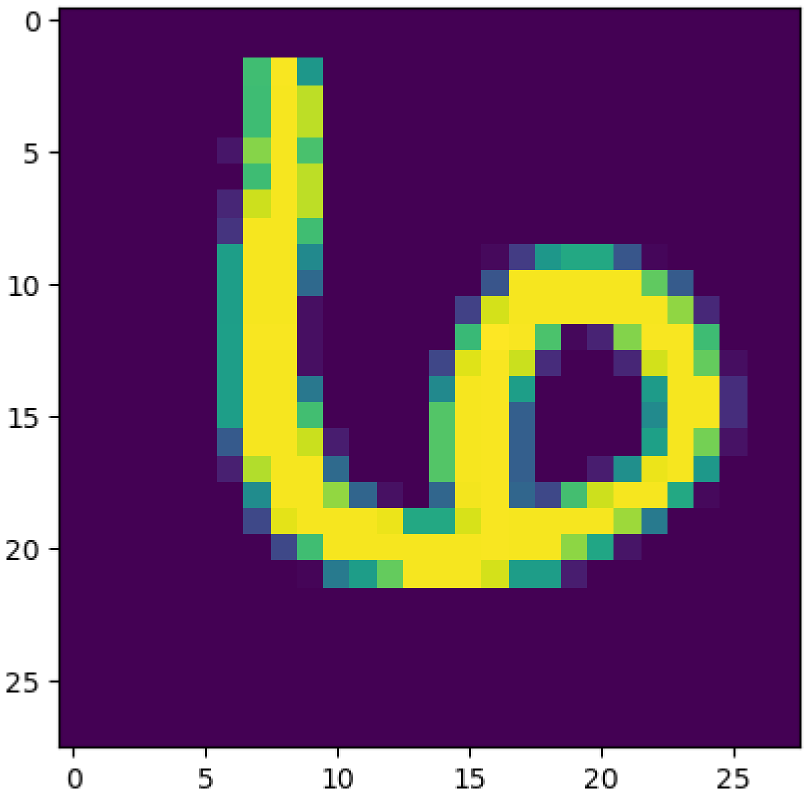
```
Epoch 1/10
1875/1875 [==============================] - 11s 5ms/step - loss: 0.6442 -
accuracy: 0.8329 - val_loss: 0.3614 - val_accuracy: 0.9022
Epoch 2/10
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3382 -
accuracy: 0.9056 - val_loss: 0.2947 - val_accuracy: 0.9183
Epoch 3/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2901 -
accuracy: 0.9183 - val_loss: 0.2625 - val_accuracy: 0.9261
Epoch 4/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2596 -
accuracy: 0.9268 - val_loss: 0.2419 - val_accuracy: 0.9309
Epoch 5/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2364 -
accuracy: 0.9334 - val_loss: 0.2241 - val_accuracy: 0.9361
Epoch 6/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2175 -
accuracy: 0.9388 - val_loss: 0.2068 - val_accuracy: 0.9415
Epoch 7/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2017 -
accuracy: 0.9431 - val_loss: 0.1945 - val_accuracy: 0.9447
Epoch 8/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.1884 -
accuracy: 0.9473 - val_loss: 0.1811 - val_accuracy: 0.9476
Epoch 9/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.1763 -
accuracy: 0.9507 - val_loss: 0.1711 - val_accuracy: 0.9504
Epoch 10/10
1875/1875 [==============================] - 5s 3ms/step - loss: 0.1662 -
accuracy: 0.9532 - val_loss: 0.1624 - val_accuracy: 0.9542
```

In [9]:
```
test_loss,test_acc=model.evaluate(x_test,y_test)
print("Loss=%.3f" %test_loss)
print("Accuracy=%.3f" %test_acc)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.1624 - ac
curacy: 0.9542
Loss=0.162
Accuracy=0.954
```

In [10]:

```
n=random.randint(0,9999)
plt.imshow(x_test[n])
plt.show()
```



In [11]:
```
x_train
```

Out[11]:
```
array([[[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., 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., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]]])
```

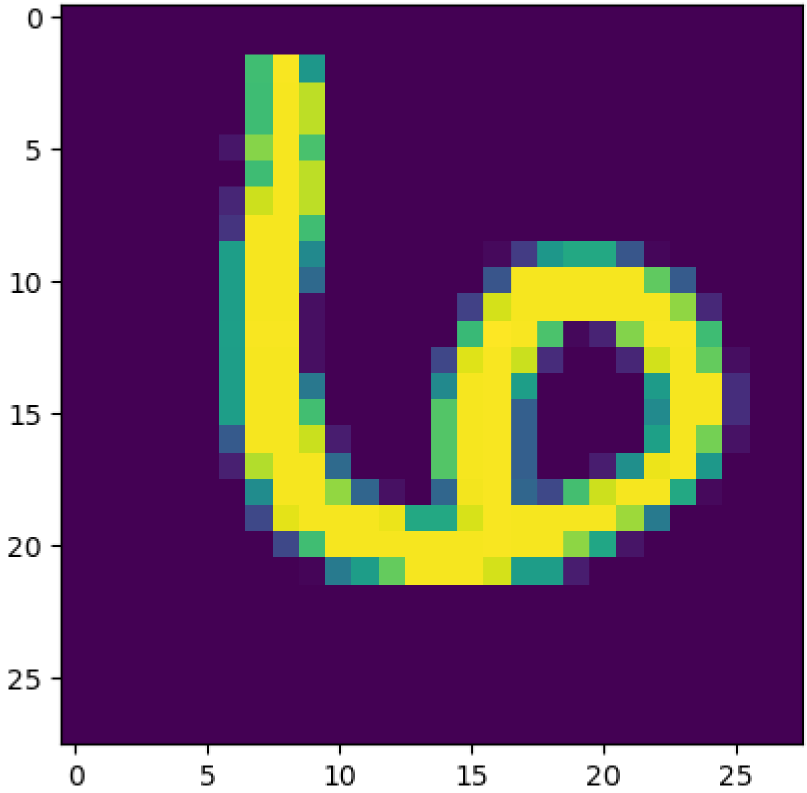
In [12]:
```
x_test
```

Out[12]:
```
array([[[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., 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., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]]])
```

In [13]:

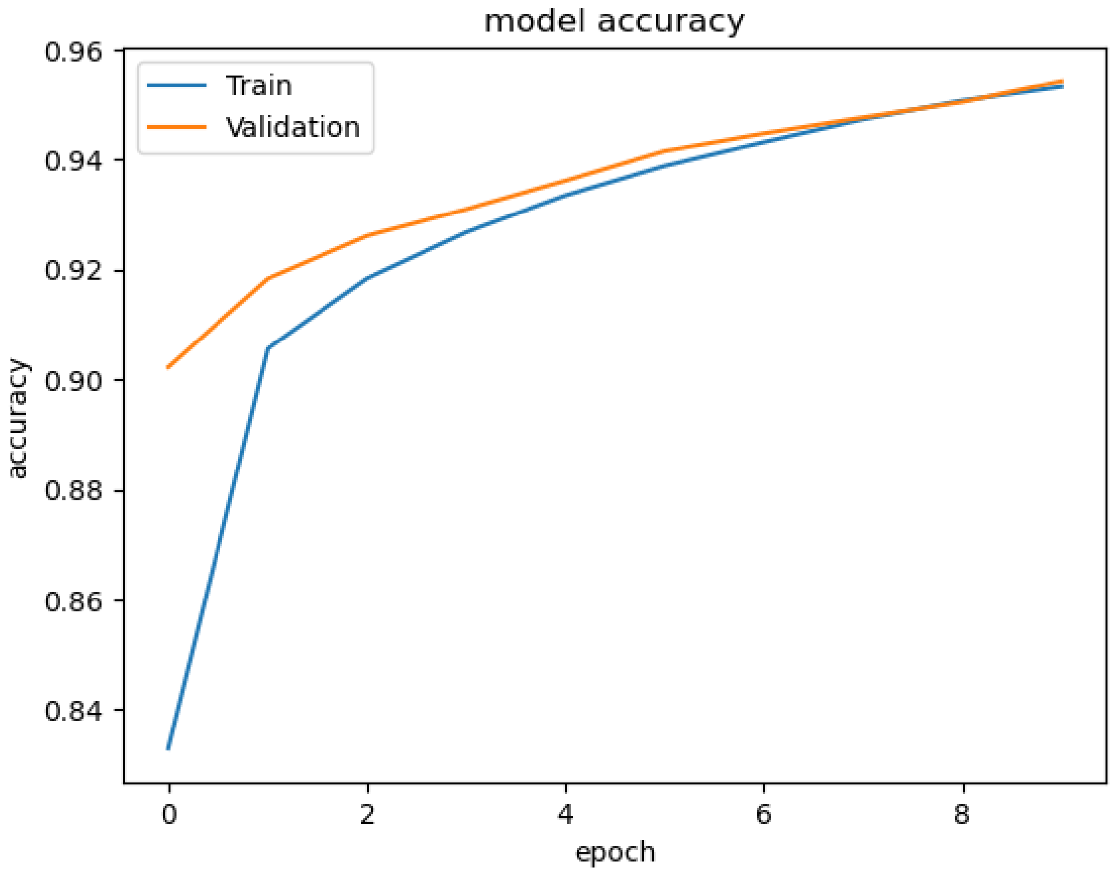
```
predicted_value=model.predict(x_test)
plt.imshow(x_test[n])
plt.show()

print(predicted_value[n])
```

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



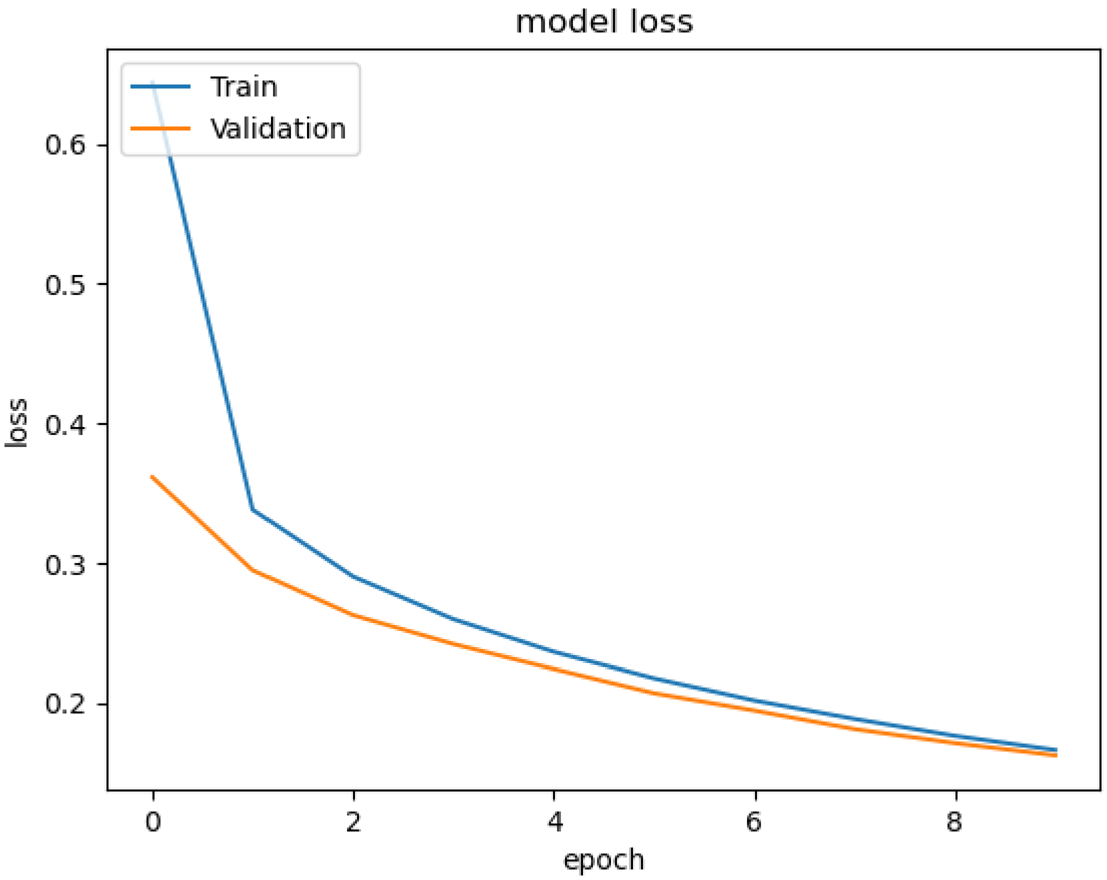
```
[8.5052976e-05 2.3059032e-07 6.5119943e-04 5.7938705e-06 3.3232808e-04
 1.3227483e-06 9.9884486e-01 6.6314547e-05 7.6497181e-06 5.2486216e-06]
```

In [14]:
```
history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```



In [15]:

```
# history.history()
history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```



In [ ]: