```python
import pandas as pd
import numpy as np

from tensorflow.keras.datasets import mnist

df=mnist.load_data()

df
```

```
((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]]], dtype=uint8),
  array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)),
 (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]]], dtype=uint8),
  array([7, 2, 1, ..., 4, 5, 6], dtype=uint8)))
```

```
(xtrain,ytrain),(xtest,ytest)=df
```

```
xtrain.shape,xtest.shape
```

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

```
ytrain.shape,ytest.shape
```

```
((60000,), (10000,))
```

```
ytrain
```

```
array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)
```

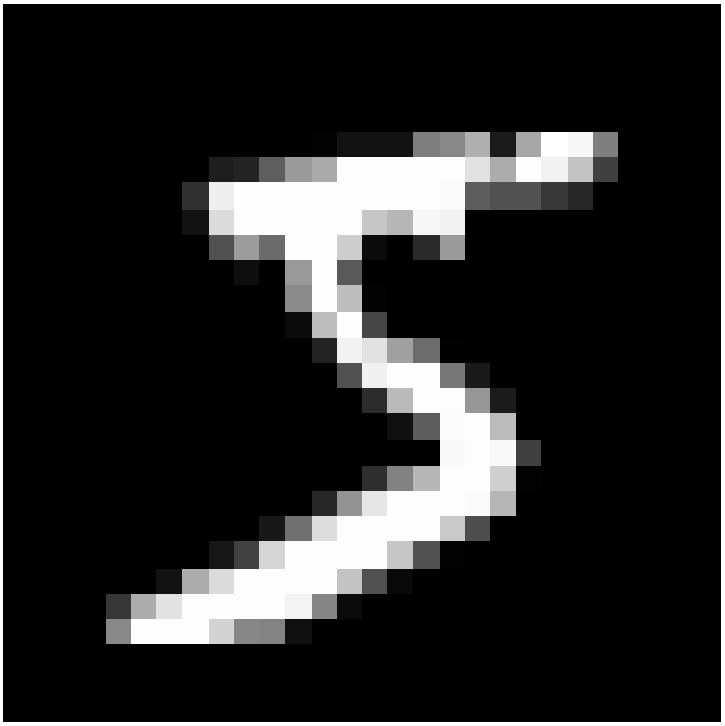
```
ytest
```

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

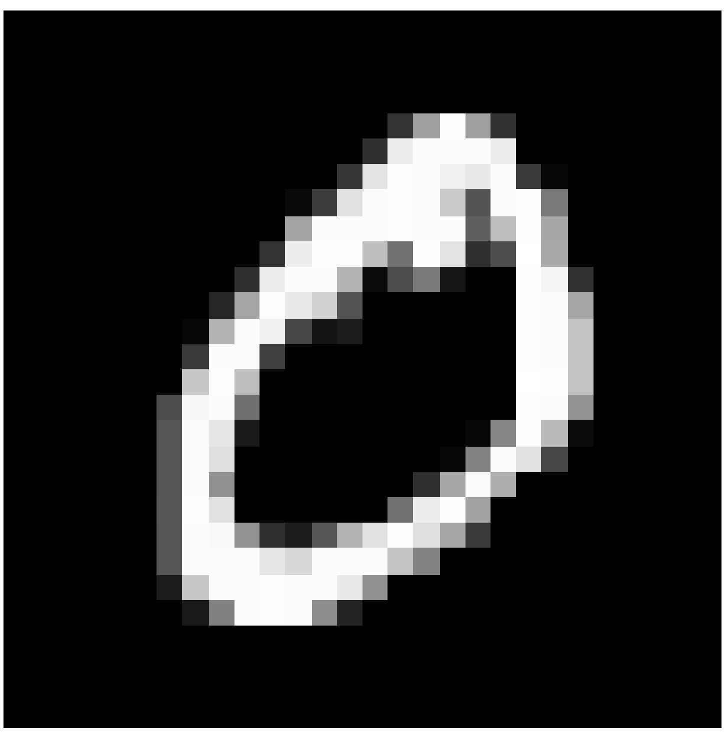
Visualizing the number from datasets

```
import matplotlib.pyplot as plt

# Function to display a single image
def visualize_image(index):
    plt.imshow(xtrain[index], cmap='gray')
    plt.title(f"Label: {ytrain[index]}")
    plt.axis('off')
    plt.show()

# Visualize the first 10 images from the dataset
for i in range(10):
    visualize_image(i)
```

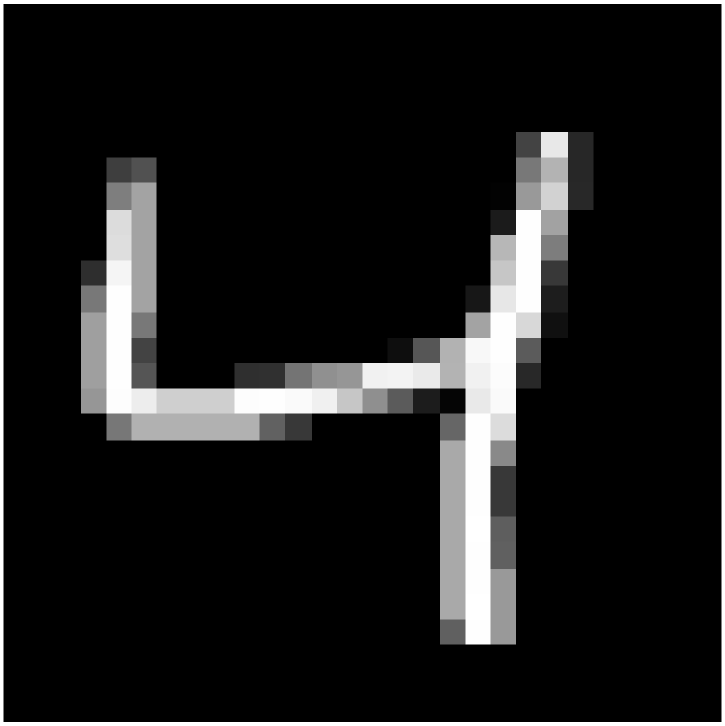
Label: 5



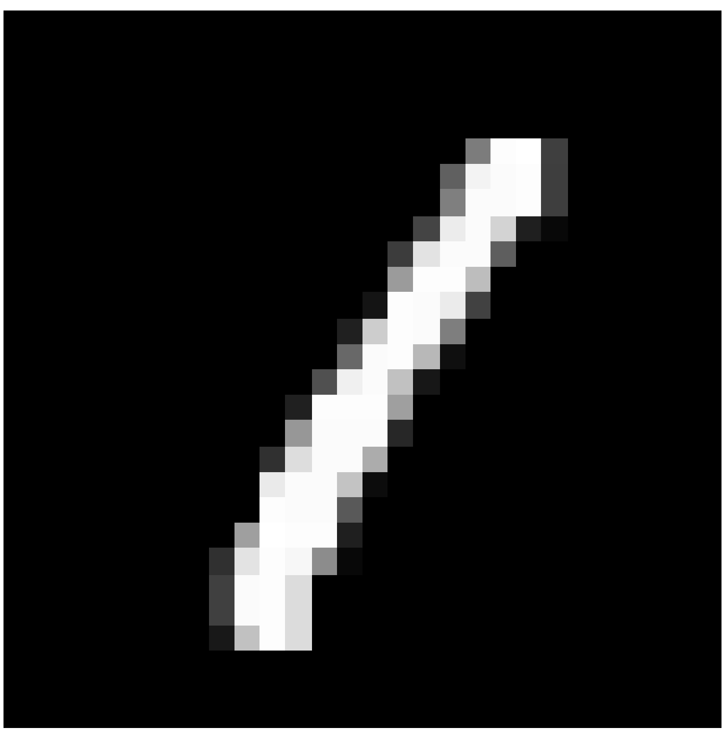
Label: 0



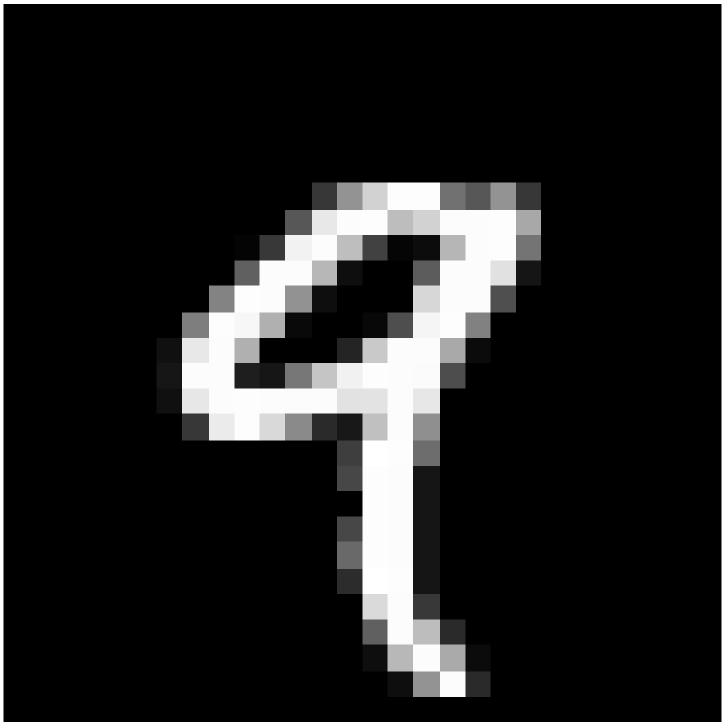
Label: 4



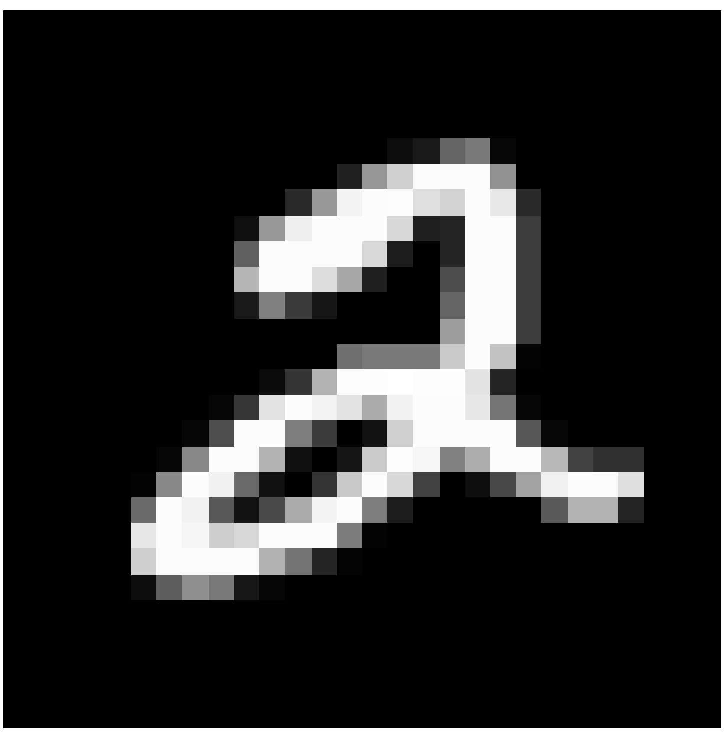
Label: 1



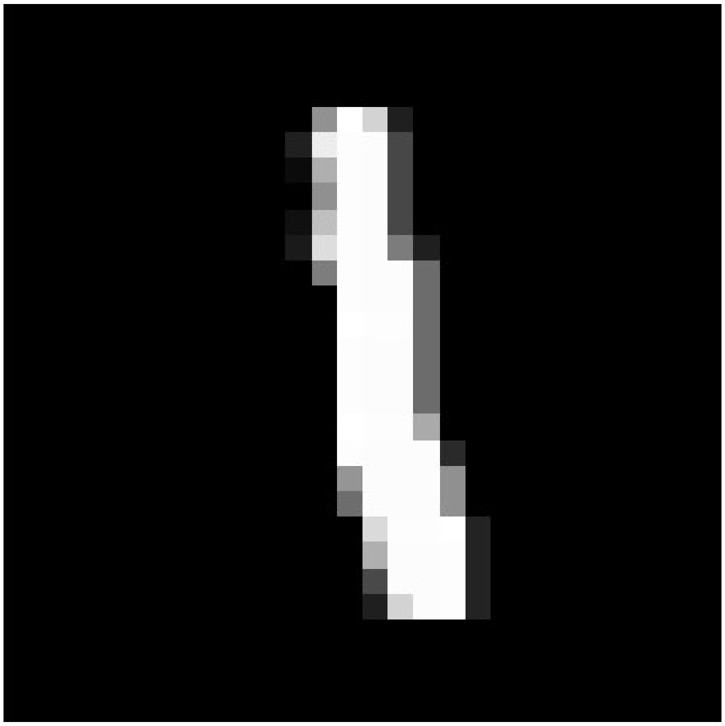
Label: 9



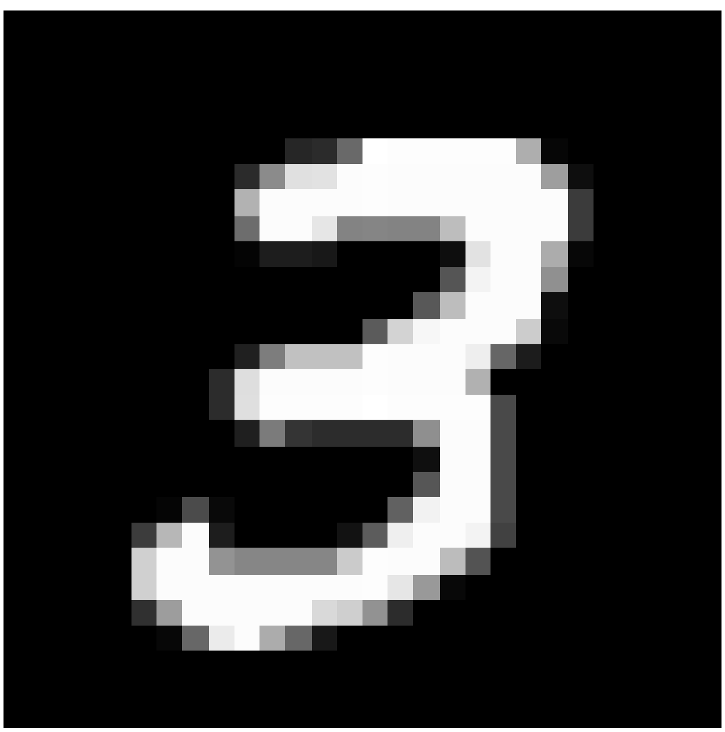
Label: 2



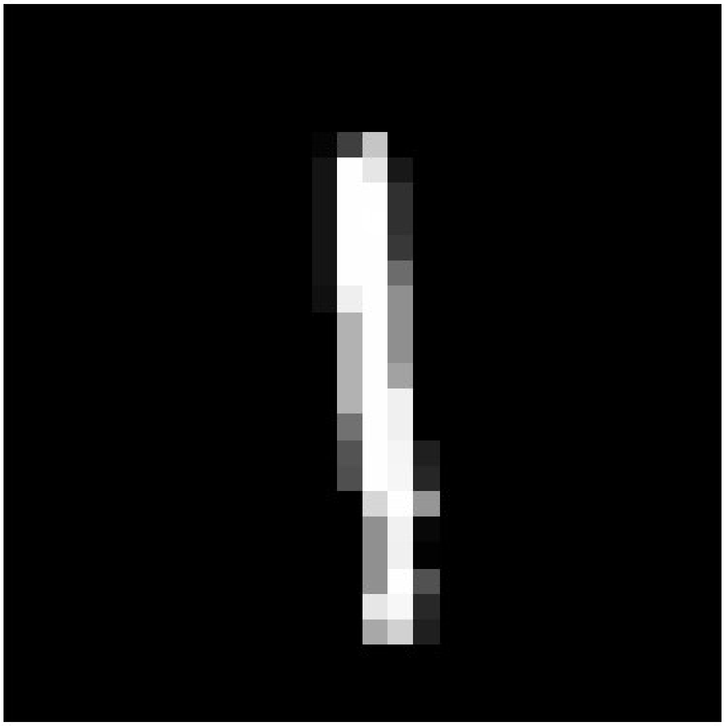
Label: 1



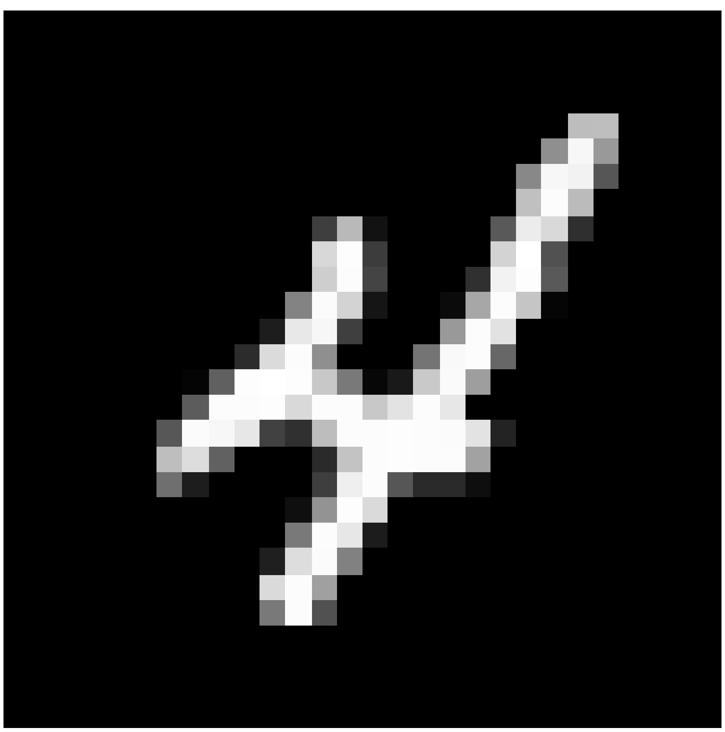
Label: 3



Label: 1



Label: 4



```
ytrain.dtype,ytest.dtype
```

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

```python
ytrain=ytrain.astype(int)
ytest=ytest.astype(int)
```

```python
ytrain.dtype,ytest.dtype
```

```
(dtype('int64'), dtype('int64'))
```

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
```

```python
model=Sequential()
```

```python
#activation function relu ;
model.add(Dense(128,input_shape=(28,28,),activation='relu'))
model.add(Dense(64,activation='relu')) #1st hidden layer
model.add(Flatten())
model.add(Dense(10,activation='softmax'))
```

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

```python
model.fit(xtrain,ytrain,epochs=50)
```

```python
model.evaluate(xtrain,ytrain)
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 6s 3ms/step - accuracy: 0.9871 - loss:
0.0649
```

```
[0.06900202482938766, 0.9865166544914246]
```

```python
y1=model.predict(xtest)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```

```python
y1
```

```
array([[0.0000000e+00, 0.0000000e+00, 2.8224962e-35, ..., 9.9999994e-
01,
         0.0000000e+00, 1.1337946e-34],
       [0.0000000e+00, 1.8604323e-20, 9.9999994e-01, ...,
0.0000000e+00,
         4.0501794e-16, 0.0000000e+00],
       [0.0000000e+00, 9.9999994e-01, 1.0149041e-21, ..., 9.3135097e-
34,
         0.0000000e+00, 0.0000000e+00],
       ...,
       [0.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 4.8835472e-
24,
         8.9962360e-18, 3.9227461e-20],
       [5.0459558e-38, 0.0000000e+00, 0.0000000e+00, ...,
```

```
0.0000000e+00,
        1.1419020e-09, 0.0000000e+00],
       [0.0000000e+00, 0.0000000e+00, 0.0000000e+00, ...,
0.0000000e+00,
        0.0000000e+00, 0.0000000e+00]], dtype=float32)
```

```
ypred1=[np.argmax(element) for element in y1]
```

```
ypred1
```

```
[7,
 2,
 1,
 0,
 4,
 1,
 4,
 9,
 6,
 9,
 0,
 6,
 9,
 0,
 1,
 5,
 9,
 7,
 3,
 4,
 9,
 6,
 6,
 5,
 4,
 0,
 7,
 4,
 0,
 1,
 3,
 1,
 3,
 4,
 7,
 2,
 7,
 1,
 3,
 1,
```

```
1,
7,
4,
2,
3,
5,
1,
2,
4,
4,
6,
3,
5,
5,
6,
0,
4,
1,
9,
5,
7,
2,
9,
3,
7,
4,
6,
4,
3,
0,
7,
0,
2,
9,
1,
7,
3,
2,
9,
7,
7,
6,
2,
7,
8,
4,
7,
3,
6,
```

```
1,
3,
6,
9,
3,
1,
4,
1,
7,
5,
9,
6,
0,
5,
4,
9,
9,
2,
1,
9,
4,
8,
1,
3,
9,
7,
4,
4,
4,
7,
2,
5,
4,
7,
6,
7,
9,
0,
5,
8,
5,
6,
6,
5,
7,
8,
1,
0,
1,
```

```
6,
4,
6,
7,
3,
1,
7,
1,
8,
2,
0,
4,
9,
8,
5,
5,
1,
5,
6,
0,
3,
4,
4,
6,
5,
4,
6,
5,
4,
5,
1,
4,
4,
7,
2,
3,
2,
7,
1,
8,
1,
8,
1,
8,
5,
0,
8,
9,
2,
```

```
3,
0,
1,
1,
1,
0,
9,
0,
3,
1,
6,
4,
3,
3,
6,
1,
1,
1,
3,
9,
5,
2,
9,
4,
5,
9,
3,
9,
0,
3,
6,
5,
5,
7,
2,
2,
7,
1,
2,
8,
4,
1,
7,
3,
3,
8,
8,
7,
7,
```

```
2,
2,
4,
1,
5,
9,
9,
7,
2,
3,
0,
4,
4,
2,
4,
1,
9,
5,
7,
7,
2,
8,
2,
6,
8,
5,
7,
7,
9,
1,
6,
1,
8,
0,
3,
0,
1,
9,
9,
4,
1,
8,
2,
1,
2,
9,
7,
5,
9,
```

```
2,
6,
4,
1,
5,
8,
2,
9,
2,
0,
4,
0,
0,
2,
8,
4,
7,
1,
2,
4,
0,
2,
7,
4,
3,
3,
0,
0,
3,
1,
9,
6,
5,
3,
5,
9,
7,
9,
3,
0,
4,
6,
0,
7,
1,
1,
2,
1,
5,
```

```
3,
3,
9,
7,
8,
6,
3,
4,
1,
3,
8,
1,
0,
5,
1,
3,
1,
5,
5,
6,
1,
8,
5,
1,
7,
4,
4,
6,
2,
2,
5,
0,
6,
5,
6,
3,
7,
2,
0,
8,
8,
5,
9,
1,
1,
4,
0,
7,
3,
```

```
7,
6,
1,
6,
2,
1,
9,
2,
0,
6,
1,
9,
5,
2,
5,
4,
4,
2,
8,
3,
8,
2,
4,
5,
0,
3,
1,
7,
7,
5,
7,
4,
7,
1,
9,
2,
1,
4,
2,
9,
2,
0,
4,
9,
1,
4,
8,
1,
8,
```

```
4,
5,
9,
3,
8,
3,
7,
6,
0,
0,
3,
0,
2,
6,
6,
4,
9,
3,
3,
3,
2,
3,
9,
1,
2,
6,
8,
0,
5,
6,
6,
6,
3,
8,
8,
2,
3,
5,
8,
9,
6,
1,
8,
4,
1,
2,
5,
3,
1,
```

```
9,
7,
5,
4,
0,
8,
9,
9,
1,
0,
5,
2,
3,
7,
8,
9,
4,
0,
6,
3,
9,
3,
2,
1,
3,
1,
3,
6,
5,
7,
4,
2,
2,
6,
3,
2,
6,
5,
4,
8,
9,
7,
1,
3,
0,
3,
8,
3,
1,
9,
```

```
3,
4,
4,
6,
4,
2,
1,
8,
2,
5,
4,
8,
8,
4,
0,
0,
2,
3,
2,
7,
1,
0,
8,
7,
4,
4,
7,
9,
6,
9,
0,
9,
8,
0,
4,
6,
0,
6,
3,
5,
4,
8,
3,
3,
9,
3,
3,
7,
7,
```

```
8,
0,
2,
2,
1,
7,
0,
6,
5,
4,
3,
6,
0,
9,
6,
3,
8,
0,
9,
9,
6,
8,
6,
8,
5,
7,
8,
6,
0,
2,
4,
0,
2,
2,
3,
1,
9,
2,
5,
1,
0,
8,
4,
6,
2,
6,
7,
7,
9,
```

```
2,
9,
8,
2,
2,
9,
2,
7,
3,
5,
9,
1,
8,
0,
2,
0,
5,
6,
1,
3,
7,
6,
7,
1,
2,
5,
8,
0,
3,
7,
1,
4,
0,
9,
1,
8,
6,
9,
7,
4,
3,
4,
9,
1,
9,
5,
1,
7,
3,
```

```
9,
7,
6,
9,
1,
3,
3,
8,
3,
3,
6,
4,
2,
8,
5,
8,
5,
1,
1,
4,
4,
3,
1,
0,
7,
7,
0,
7,
9,
4,
4,
8,
5,
5,
4,
0,
8,
2,
1,
6,
8,
4,
5,
0,
4,
0,
6,
1,
7,
```

```
3,
2,
6,
7,
2,
6,
9,
3,
1,
4,
6,
2,
5,
9,
2,
0,
6,
2,
1,
7,
3,
4,
1,
0,
5,
4,
3,
1,
1,
7,
4,
9,
9,
4,
8,
4,
0,
2,
4,
5,
1,
1,
6,
4,
7,
1,
9,
4,
2,
```

```
4,
1,
5,
5,
3,
8,
3,
1,
4,
5,
6,
8,
9,
9,
1,
9,
3,
8,
0,
3,
2,
5,
1,
2,
8,
3,
4,
4,
0,
8,
8,
3,
3,
1,
7,
3,
5,
9,
6,
3,
2,
6,
1,
3,
6,
0,
7,
2,
1,
```

```
7,
1,
4,
2,
4,
2,
1,
7,
9,
6,
1,
1,
2,
4,
8,
1,
7,
7,
4,
7,
5,
7,
3,
1,
3,
1,
0,
7,
7,
0,
3,
5,
5,
2,
7,
6,
6,
9,
2,
8,
3,
5,
2,
2,
5,
6,
0,
8,
2,
```

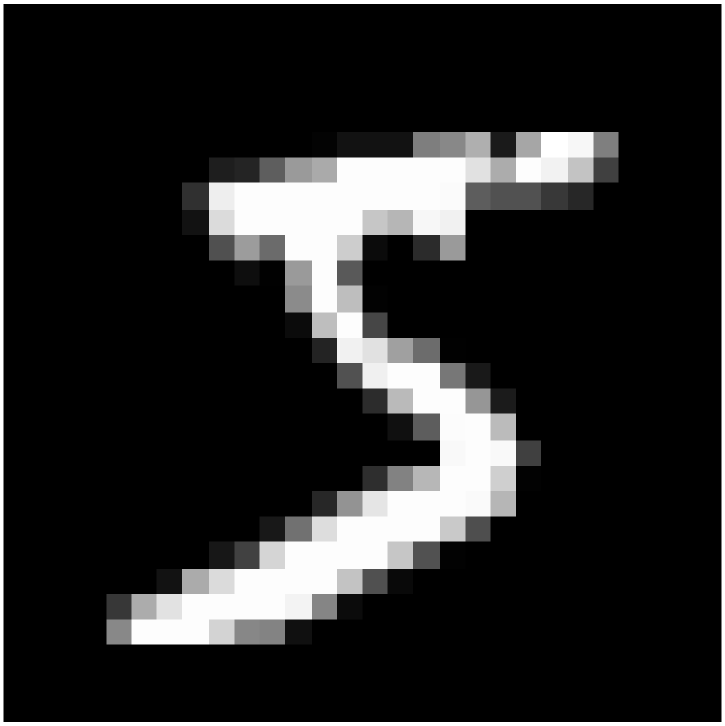
```
9,
2,
8,
2,
8,
8,
7,
4,
7,
5,
0,
6,
6,
3,
2,
1,
5,
2,
2,
9,
3,
0,
0,
5,
7,
8,
1,
4,
4,
6,
0,
2,
7,
1,
4,
7,
4,
7,
3,
9,
8,
8,
4,
7,
1,
2,
1,
2,
2,
```

```
3,
3,
3,
2,
3,
9,
1,
7,
4,
0,
3,
5,
5,
8,
6,
3,
9,
6,
7,
6,
6,
3,
2,
7,
8,
1,
1,
7,
4,
6,
4,
9,
5,
1,
9,
3,
4,
7,
8,
9,
1,
1,
6,
9,
1,
4,
4,
5,
4,
```

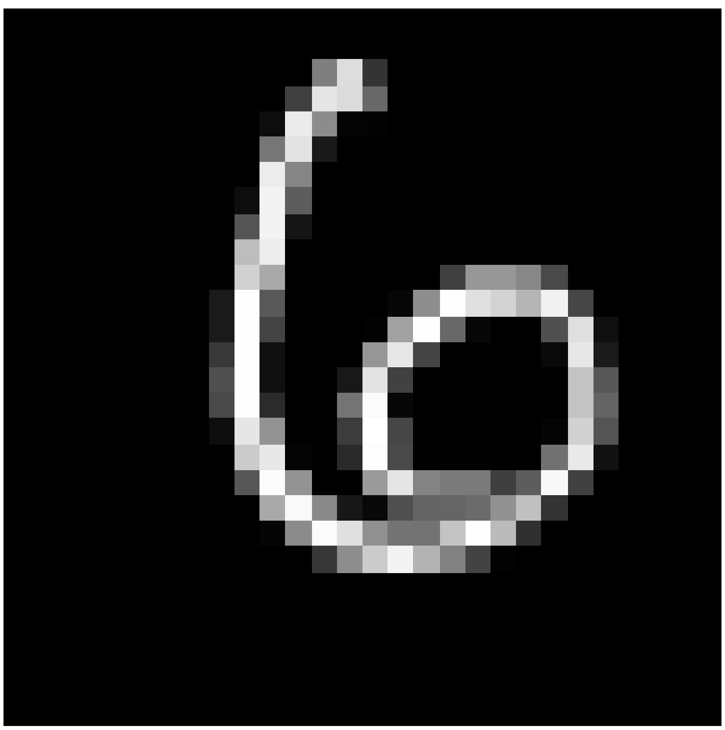
```
 0,
 6,
 2,
 2,
 3,
 1,
 5,
 1,
 2,
 0,
 3,
 8,
 1,
 2,
 6,
 7,
 1,
 6,
 3,
 3,
 9,
 0,
 1,
 2,
 2,
 0,
 8,
 7,
 ...]
```

```python
def visualize_image(index, dataset='train'):
    if dataset == 'train':
        image_data = xtrain[index]
        label = ytrain[index]
    elif dataset == 'test':
        image_data = xtest[index]
        label = ypred1[index]
    else:
        print("Invalid dataset. Choose 'train' or 'test'.")
        return

    plt.imshow(image_data, cmap='gray')
    plt.title(f"Label: {label}")
    plt.axis('off')
    plt.show()

# Example usage:
visualize_image(0, 'train')
visualize_image(100, 'test')
```

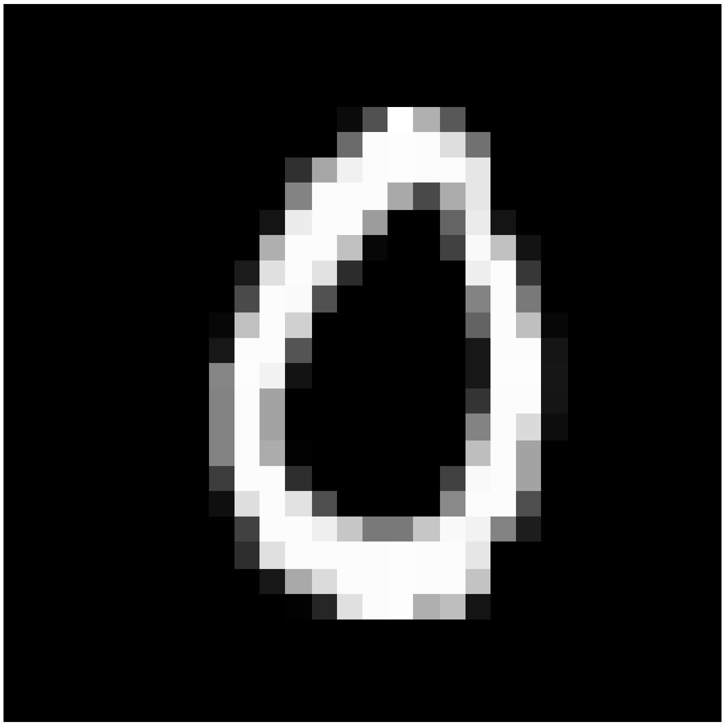
Label: 5



Label: 6



```
visualize_image(1001,'test')
```




```
#activation function tanh
model2=Sequential()

model2.add(Dense(64,activation='tanh',input_shape=(28,28,)))
model2.add(Dense(168,activation='tanh'))
model2.add(Flatten())
model2.add(Dense(10,activation='softmax'))

model2.compile(optimizer='adam',loss='sparse_categorical_crossentropy'
,metrics=['accuracy'])

model2.fit(xtrain,ytrain,epochs=50)

Epoch 1/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 15s 7ms/step - accuracy: 0.8838 - loss:
0.3740
Epoch 2/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9431 - loss:
0.1889
Epoch 3/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9507 - loss:
0.1636
Epoch 4/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9535 - loss:
0.1607
Epoch 5/50
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9538 - loss:
0.1565
Epoch 6/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9582 - loss:
0.1429
Epoch 7/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9597 - loss:
0.1381
Epoch 8/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 8ms/step - accuracy: 0.9591 - loss:
0.1400
Epoch 9/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9607 - loss:
0.1346
Epoch 10/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9643 - loss:
0.1238
Epoch 11/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9655 - loss:
0.1182
Epoch 12/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9641 - loss:
0.1190
Epoch 13/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 8ms/step - accuracy: 0.9656 - loss:
0.1110
Epoch 14/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9664 - loss:
0.1156
Epoch 15/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9697 - loss:
0.1075
Epoch 16/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9677 - loss:
0.1090
Epoch 17/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 8ms/step - accuracy: 0.9684 - loss:
0.1047
Epoch 18/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9703 - loss:
0.0994
Epoch 19/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9711 - loss:
0.0963
Epoch 20/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9702 - loss:
0.0970
Epoch 21/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9667 - loss:
```

```
0.1082
Epoch 22/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9698 - loss:
0.1018
Epoch 23/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9710 - loss:
0.1002
Epoch 24/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9713 - loss:
0.0936
Epoch 25/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9711 - loss:
0.0981
Epoch 26/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9705 - loss:
0.1013
Epoch 27/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 8ms/step - accuracy: 0.9724 - loss:
0.0950
Epoch 28/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9729 - loss:
0.0902
Epoch 29/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 8ms/step - accuracy: 0.9725 - loss:
0.0921
Epoch 30/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 8ms/step - accuracy: 0.9752 - loss:
0.0819
Epoch 31/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9733 - loss:
0.0910
Epoch 32/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9745 - loss:
0.0905
Epoch 33/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9754 - loss:
0.0868
Epoch 34/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9738 - loss:
0.0866
Epoch 35/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9720 - loss:
0.0976
Epoch 36/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9750 - loss:
0.0815
Epoch 37/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9771 - loss:
0.0765
```

```
Epoch 38/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9760 - loss:
0.0812
Epoch 39/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9772 - loss:
0.0821
Epoch 40/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9756 - loss:
0.0843
Epoch 41/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9751 - loss:
0.0869
Epoch 42/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9761 - loss:
0.0827
Epoch 43/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9771 - loss:
0.0795
Epoch 44/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9777 - loss:
0.0806
Epoch 45/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9765 - loss:
0.0821
Epoch 46/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 8ms/step - accuracy: 0.9771 - loss:
0.0793
Epoch 47/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 8ms/step - accuracy: 0.9759 - loss:
0.0853
Epoch 48/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 8ms/step - accuracy: 0.9763 - loss:
0.0843
Epoch 49/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9773 - loss:
0.0803
Epoch 50/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 8ms/step - accuracy: 0.9766 - loss:
0.0800

<keras.src.callbacks.history.History at 0x784f88c20640>
```

```
model2.evaluate(xtrain,ytrain)
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 7s 4ms/step - accuracy: 0.9804 - loss:
0.0685

[0.06562913209199905, 0.981166660785675]
```

```
y2=model2.predict(xtest)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step
```

```
y2
```

```
array([[1.4036738e-36, 2.7824912e-21, 2.8359143e-18, ..., 9.9999994e-
01,
        5.2625744e-25, 2.1223770e-15],
       [1.2844649e-17, 1.4631690e-13, 9.9999994e-01, ..., 6.5610199e-
36,
        1.2864052e-08, 1.4660928e-26],
       [5.6874360e-35, 9.9999994e-01, 2.5843178e-10, ..., 9.0244637e-
16,
        9.9850897e-17, 1.3614328e-20],
       ...,
       [4.4535146e-31, 1.0835142e-26, 8.6005205e-19, ..., 4.8715567e-
12,
        3.7982208e-09, 7.3332991e-08],
       [4.6053534e-09, 8.4573046e-20, 1.0897724e-14, ..., 7.4114695e-
16,
        1.4756590e-05, 8.5103604e-17],
       [1.7599183e-19, 8.5153641e-23, 4.5789701e-20, ..., 1.6007687e-
33,
        1.5706022e-20, 2.6138973e-22]], dtype=float32)
```

```
ypred2=[np.argmax(element) for element in y2]
```

```
ypred2
```

```
[7,
 2,
 1,
 0,
 4,
 1,
 4,
 9,
 6,
 9,
 0,
 6,
 9,
 0,
 1,
 5,
 9,
 7,
 3,
 4,
 9,
 6,
```

```
6,
5,
4,
0,
7,
4,
0,
1,
3,
1,
3,
4,
7,
2,
7,
1,
2,
1,
1,
7,
4,
2,
3,
5,
1,
2,
4,
4,
6,
3,
5,
5,
6,
0,
4,
1,
9,
5,
7,
8,
9,
3,
7,
4,
6,
4,
3,
0,
7,
```

```
0,
2,
9,
1,
7,
3,
2,
9,
7,
7,
6,
2,
7,
8,
4,
7,
3,
6,
1,
3,
6,
9,
3,
1,
4,
1,
7,
6,
9,
6,
0,
5,
4,
9,
9,
2,
1,
9,
4,
8,
7,
3,
9,
7,
4,
4,
4,
9,
2,
```

```
5,
4,
7,
6,
7,
9,
0,
5,
8,
5,
6,
6,
5,
7,
8,
1,
0,
1,
6,
4,
6,
7,
3,
1,
7,
1,
8,
2,
0,
3,
9,
9,
5,
5,
1,
5,
6,
0,
3,
4,
4,
6,
5,
4,
6,
5,
4,
5,
1,
```

```
4,
4,
7,
2,
3,
2,
7,
1,
8,
1,
8,
1,
8,
5,
0,
8,
9,
2,
5,
0,
1,
1,
1,
0,
9,
0,
3,
1,
6,
4,
3,
3,
6,
1,
1,
1,
3,
9,
5,
2,
9,
4,
5,
9,
3,
9,
0,
3,
6,
```

```
5,
5,
7,
2,
2,
7,
1,
2,
8,
4,
1,
7,
3,
3,
8,
8,
7,
9,
2,
2,
4,
1,
5,
9,
8,
7,
2,
3,
0,
6,
4,
2,
4,
1,
9,
5,
7,
7,
2,
8,
2,
6,
8,
5,
7,
7,
9,
1,
8,
```

```
1,
8,
0,
3,
0,
1,
9,
9,
4,
1,
8,
2,
1,
2,
9,
3,
5,
9,
2,
6,
4,
1,
5,
8,
2,
9,
2,
0,
4,
0,
0,
2,
8,
4,
7,
1,
2,
4,
0,
2,
7,
4,
3,
3,
0,
0,
3,
1,
9,
```

```
6,
5,
3,
5,
1,
7,
9,
3,
0,
4,
2,
0,
7,
1,
1,
2,
1,
5,
3,
3,
9,
7,
8,
6,
3,
6,
1,
3,
8,
1,
0,
5,
1,
3,
1,
5,
8,
6,
1,
8,
5,
1,
3,
9,
4,
6,
2,
2,
5,
```

```
0,
6,
5,
6,
3,
7,
2,
0,
8,
8,
5,
4,
1,
1,
4,
0,
7,
3,
7,
6,
1,
6,
2,
1,
9,
2,
8,
6,
1,
9,
5,
2,
5,
4,
4,
2,
8,
3,
8,
2,
4,
9,
0,
3,
1,
7,
7,
5,
7,
```

```
9,
7,
1,
7,
2,
1,
4,
2,
9,
2,
0,
4,
9,
1,
4,
8,
1,
8,
4,
5,
9,
8,
8,
3,
7,
6,
0,
0,
3,
0,
2,
0,
6,
9,
8,
5,
3,
3,
2,
3,
7,
1,
2,
6,
8,
0,
5,
6,
6,
```

```
6,
3,
8,
8,
2,
7,
5,
8,
9,
6,
1,
8,
4,
1,
2,
3,
9,
1,
9,
7,
5,
4,
0,
8,
9,
9,
1,
0,
5,
2,
3,
7,
2,
9,
4,
0,
6,
3,
9,
3,
2,
1,
3,
1,
3,
6,
5,
7,
4,
2,
```

```
2,
6,
3,
2,
6,
5,
4,
8,
9,
7,
1,
3,
0,
3,
8,
3,
1,
9,
2,
4,
4,
6,
4,
1,
1,
8,
2,
5,
4,
8,
8,
4,
0,
0,
2,
3,
2,
7,
1,
0,
8,
7,
4,
4,
7,
9,
6,
9,
0,
```

```
9,
8,
0,
4,
6,
0,
6,
3,
5,
4,
8,
3,
3,
9,
3,
3,
3,
7,
8,
0,
8,
8,
1,
7,
0,
6,
5,
4,
3,
8,
0,
9,
6,
3,
8,
0,
9,
9,
6,
8,
6,
8,
5,
7,
8,
6,
0,
2,
6,
```

```
0,
2,
2,
3,
1,
9,
7,
5,
1,
0,
8,
4,
6,
2,
6,
7,
9,
9,
2,
9,
8,
2,
2,
9,
2,
7,
3,
5,
9,
1,
8,
0,
2,
0,
5,
6,
1,
3,
7,
6,
7,
1,
2,
5,
8,
0,
3,
7,
8,
```

```
4,
0,
9,
1,
8,
6,
9,
7,
4,
3,
4,
9,
1,
9,
5,
1,
7,
3,
9,
7,
6,
9,
1,
3,
7,
8,
3,
3,
6,
4,
2,
4,
5,
8,
8,
1,
1,
4,
4,
3,
1,
0,
7,
7,
0,
7,
9,
4,
4,
```

```
8,
5,
5,
4,
0,
8,
2,
1,
6,
8,
4,
8,
0,
4,
0,
6,
1,
7,
3,
2,
6,
7,
2,
6,
9,
3,
1,
4,
6,
2,
5,
9,
2,
0,
6,
2,
1,
7,
3,
4,
1,
0,
5,
4,
3,
1,
1,
7,
4,
```

```
9,
9,
4,
8,
4,
0,
2,
4,
5,
1,
1,
6,
4,
7,
1,
9,
4,
2,
4,
1,
5,
5,
3,
8,
3,
1,
4,
5,
6,
8,
9,
4,
1,
5,
3,
8,
0,
1,
2,
5,
1,
2,
8,
3,
4,
4,
0,
8,
8,
```

```
3,
3,
1,
7,
3,
5,
9,
6,
3,
2,
6,
1,
3,
6,
0,
7,
0,
1,
7,
1,
4,
2,
4,
2,
1,
7,
9,
6,
1,
1,
2,
4,
5,
1,
7,
7,
4,
8,
0,
9,
3,
1,
3,
1,
0,
7,
7,
0,
3,
```

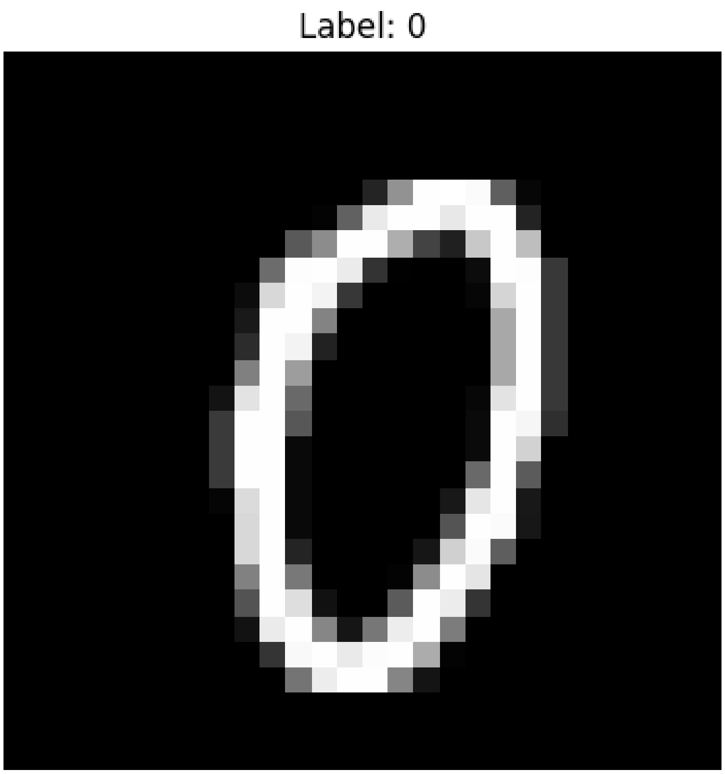
```
5,
5,
2,
7,
6,
6,
9,
2,
8,
3,
5,
2,
2,
5,
6,
0,
8,
2,
9,
2,
8,
2,
8,
8,
7,
4,
9,
3,
0,
6,
6,
3,
2,
1,
3,
2,
2,
9,
3,
0,
0,
5,
7,
8,
1,
4,
4,
6,
0,
```

```
2,
9,
1,
4,
7,
4,
7,
3,
9,
8,
8,
4,
7,
1,
2,
1,
2,
2,
3,
3,
3,
2,
3,
9,
1,
7,
4,
0,
3,
5,
5,
8,
6,
3,
2,
6,
7,
6,
6,
5,
2,
7,
8,
1,
1,
7,
4,
6,
4,
```

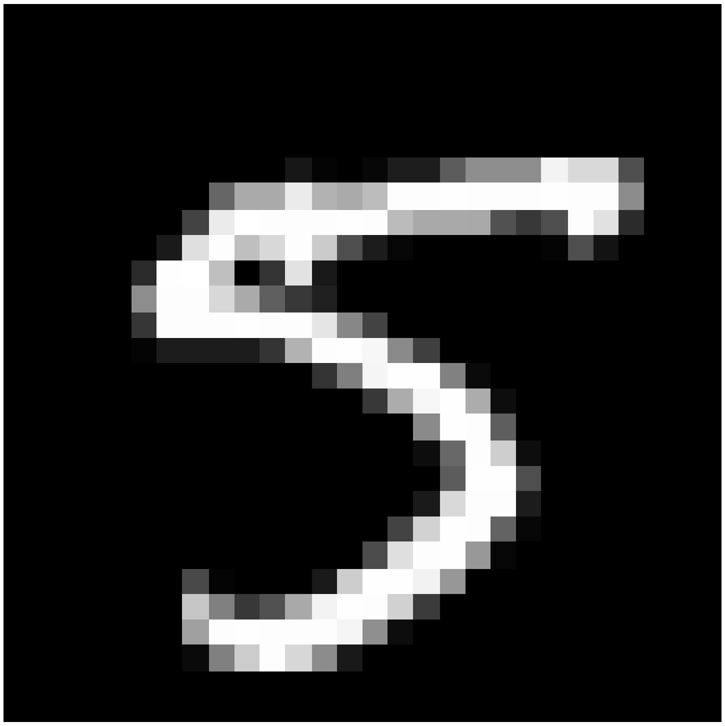
```
 9,
 3,
 2,
 3,
 3,
 4,
 7,
 8,
 9,
 1,
 1,
 0,
 9,
 1,
 4,
 4,
 5,
 4,
 0,
 6,
 2,
 2,
 3,
 1,
 5,
 1,
 2,
 0,
 2,
 8,
 1,
 2,
 6,
 7,
 1,
 6,
 2,
 3,
 4,
 0,
 1,
 2,
 2,
 0,
 8,
 9,
 ...]
```

```
def visualize_image2(index, dataset='train'):
    if dataset == 'train':
```

```
        image_data = xtrain[index]
        label = ytrain[index]
    elif dataset == 'test':
        image_data = xtest[index]
        label = ypred2[index]
    else:
        print("Invalid dataset. Choose 'train' or 'test'.")
        return

    plt.imshow(image_data, cmap='gray')
    plt.title(f"Label: {label}")
    plt.axis('off')
    plt.show()

# Example usage:
visualize_image2(1000, 'train')
visualize_image2(102, 'test')
```

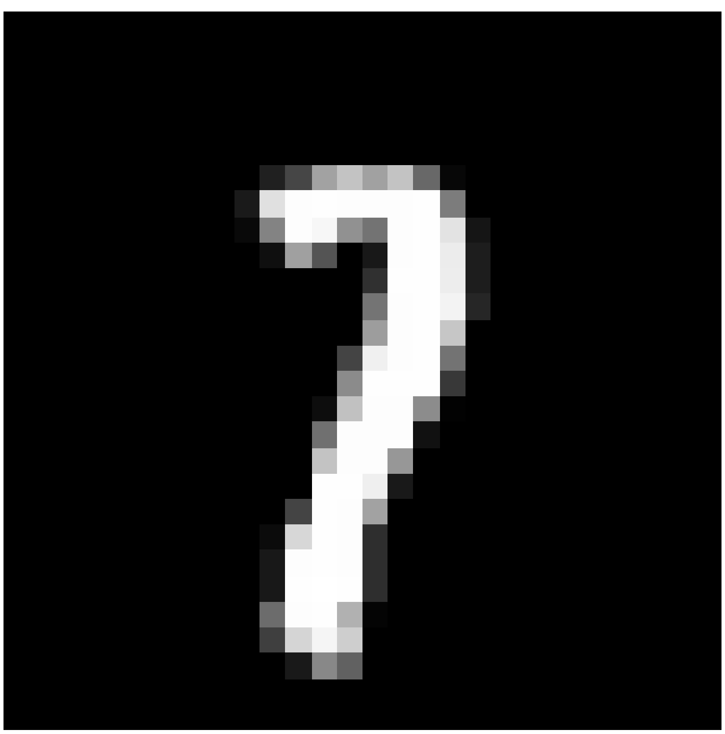


Label: 5



```
visualize_image2(111,'test')
```

Label: 7



```
#activation function sigmiod
model3=Sequential()

model3.add(Dense(128,activation='sigmoid',input_shape=(28,28,)))
model3.add(Dense(62,activation='sigmoid'))
model3.add(Flatten())
model3.add(Dense(10,activation='softmax'))

model3.compile(optimizer='adam',loss='sparse_categorical_crossentropy'
,metrics=['accuracy'])

model3.fit(xtrain,ytrain,epochs=50)
```

```
Epoch 1/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step - accuracy: 0.9934 - loss:
0.0201
Epoch 2/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9940 - loss:
0.0199
Epoch 3/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 14s 7ms/step - accuracy: 0.9931 - loss:
0.0203
Epoch 4/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step - accuracy: 0.9940 - loss:
0.0187
Epoch 5/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9935 - loss:
0.0207
Epoch 6/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9936 - loss:
0.0197
Epoch 7/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9933 - loss:
0.0205
Epoch 8/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step - accuracy: 0.9942 - loss:
0.0175
Epoch 9/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 7ms/step - accuracy: 0.9944 - loss:
0.0176
Epoch 10/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step - accuracy: 0.9949 - loss:
0.0157
Epoch 11/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9938 - loss:
0.0185
Epoch 12/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 13s 7ms/step - accuracy: 0.9948 - loss:
0.0163
Epoch 13/50
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9946 - loss: 
0.0167
Epoch 14/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9947 - loss: 
0.0164
Epoch 15/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9938 - loss: 
0.0174
Epoch 16/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9951 - loss: 
0.0159
Epoch 17/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9946 - loss: 
0.0154
Epoch 18/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9960 - loss: 
0.0132
Epoch 19/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9951 - loss: 
0.0143
Epoch 20/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9950 - loss: 
0.0146
Epoch 21/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9944 - loss: 
0.0156
Epoch 22/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9950 - loss: 
0.0155
Epoch 23/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9960 - loss: 
0.0126
Epoch 24/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9958 - loss: 
0.0129
Epoch 25/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9962 - loss: 
0.0122
Epoch 26/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9956 - loss: 
0.0126
Epoch 27/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9966 - loss: 
0.0112
Epoch 28/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9961 - loss: 
0.0119
Epoch 29/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9962 - loss: 
```

```
0.0119
Epoch 30/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9951 - loss:
0.0129
Epoch 31/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9956 - loss:
0.0132
Epoch 32/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9968 - loss:
0.0100
Epoch 33/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9966 - loss:
0.0111
Epoch 34/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9967 - loss:
0.0102
Epoch 35/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9964 - loss:
0.0114
Epoch 36/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9967 - loss:
0.0105
Epoch 37/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9967 - loss:
0.0104
Epoch 38/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9963 - loss:
0.0115
Epoch 39/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9966 - loss:
0.0108
Epoch 40/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9974 - loss:
0.0088
Epoch 41/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 22s 6ms/step - accuracy: 0.9960 - loss:
0.0117
Epoch 42/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9961 - loss:
0.0123
Epoch 43/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9964 - loss:
0.0106
Epoch 44/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9963 - loss:
0.0110
Epoch 45/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9970 - loss:
0.0099
```

```
Epoch 46/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 6ms/step - accuracy: 0.9971 - loss:
0.0095
Epoch 47/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9971 - loss:
0.0091
Epoch 48/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 11s 6ms/step - accuracy: 0.9972 - loss:
0.0089
Epoch 49/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 21s 6ms/step - accuracy: 0.9960 - loss:
0.0113
Epoch 50/50
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 12s 6ms/step - accuracy: 0.9965 - loss:
0.0099

<keras.src.callbacks.history.History at 0x784f8b14e6b0>
```

```
model3.evaluate(xtrain,ytrain)
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9973 - loss:
0.0088

[0.009506034664809704, 0.9970499873161316]
```

```
y3=model3.predict(xtest)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
```

```
y3
```

```
array([[0.00000000e+00, 1.42285323e-24, 7.77190986e-19, ...,
        9.99999940e-01, 1.11419761e-31, 1.39567290e-17],
       [6.51337354e-25, 1.84592636e-11, 9.99999344e-01, ...,
        8.29567399e-36, 5.83171289e-11, 0.00000000e+00],
       [0.00000000e+00, 9.99986231e-01, 3.48328626e-07, ...,
        1.66176710e-06, 3.01535868e-16, 7.01626259e-18],
       ...,
       [4.10492509e-33, 1.05757249e-23, 1.75383492e-16, ...,
        9.09234580e-12, 5.97551830e-09, 3.30400695e-07],
       [3.75778055e-12, 1.61500397e-18, 7.13649686e-20, ...,
        6.29282780e-18, 3.93436785e-04, 1.54809604e-16],
       [6.14351353e-27, 0.00000000e+00, 2.97564676e-23, ...,
        0.00000000e+00, 5.27243753e-26, 1.25980755e-26]],
dtype=float32)
```

```
ypred3=[np.argmax(element) for element in y3]
```

```
ypred3
```

```
[7,
 2,
```

```
1,
0,
4,
1,
4,
9,
6,
9,
0,
6,
9,
0,
1,
5,
9,
7,
3,
4,
9,
6,
6,
5,
4,
0,
7,
4,
0,
1,
3,
1,
3,
4,
7,
2,
7,
1,
2,
1,
1,
7,
4,
2,
3,
5,
1,
2,
4,
4,
6,
```

```
3,
5,
5,
6,
0,
4,
1,
9,
5,
7,
2,
9,
3,
7,
4,
6,
4,
3,
0,
7,
0,
2,
9,
1,
7,
3,
2,
9,
7,
7,
6,
2,
7,
8,
4,
7,
3,
6,
1,
3,
6,
9,
3,
1,
4,
1,
7,
6,
9,
```

```
6,
0,
5,
4,
9,
9,
2,
1,
9,
4,
8,
7,
3,
9,
7,
4,
4,
4,
9,
2,
5,
4,
7,
6,
7,
9,
0,
5,
8,
5,
6,
6,
5,
7,
8,
1,
0,
1,
6,
4,
6,
7,
3,
1,
7,
1,
8,
2,
0,
```

```
2,
9,
9,
5,
5,
1,
5,
6,
0,
3,
4,
4,
6,
5,
4,
6,
5,
4,
5,
1,
4,
4,
7,
2,
3,
2,
7,
1,
8,
1,
8,
1,
8,
5,
0,
8,
9,
2,
5,
0,
1,
1,
1,
0,
9,
0,
3,
1,
6,
```

```
4,
2,
3,
6,
1,
1,
1,
3,
9,
5,
2,
9,
4,
5,
9,
3,
9,
0,
3,
6,
5,
5,
7,
2,
2,
7,
1,
2,
8,
4,
1,
7,
3,
3,
8,
8,
7,
9,
2,
2,
4,
1,
5,
9,
8,
7,
2,
3,
0,
```

```
6,
4,
2,
4,
1,
9,
5,
7,
7,
2,
8,
2,
0,
8,
5,
7,
7,
9,
1,
8,
1,
8,
0,
3,
0,
1,
9,
9,
4,
1,
8,
2,
1,
2,
9,
7,
5,
9,
2,
6,
4,
1,
5,
8,
2,
9,
2,
0,
4,
```

```
0,
0,
2,
8,
4,
7,
1,
2,
4,
0,
2,
7,
4,
3,
3,
0,
0,
3,
1,
9,
6,
5,
2,
5,
9,
7,
9,
3,
0,
4,
2,
0,
7,
1,
1,
2,
1,
5,
3,
3,
9,
7,
8,
6,
3,
6,
1,
3,
8,
```

```
1,
0,
5,
1,
3,
1,
5,
5,
6,
1,
8,
5,
1,
3,
9,
4,
6,
2,
2,
5,
0,
6,
5,
6,
3,
7,
2,
0,
8,
8,
5,
4,
1,
1,
4,
0,
3,
3,
7,
6,
1,
6,
2,
1,
9,
2,
8,
6,
1,
```

```
9,
5,
2,
5,
4,
4,
2,
8,
3,
8,
2,
4,
5,
0,
3,
1,
7,
7,
3,
7,
9,
7,
1,
9,
2,
1,
4,
2,
9,
2,
0,
4,
9,
1,
4,
8,
1,
8,
4,
5,
9,
8,
8,
3,
7,
6,
0,
0,
3,
```

```
0,
2,
0,
6,
9,
9,
5,
3,
3,
2,
3,
9,
1,
2,
6,
8,
0,
5,
6,
6,
6,
3,
8,
8,
2,
7,
5,
8,
9,
6,
1,
8,
4,
1,
2,
5,
9,
1,
9,
7,
5,
4,
0,
8,
9,
9,
1,
0,
5,
2,
```

```
3,
7,
8,
9,
4,
0,
6,
3,
9,
5,
2,
1,
3,
1,
3,
6,
5,
7,
4,
2,
2,
6,
3,
2,
6,
5,
4,
8,
9,
7,
1,
3,
0,
3,
8,
3,
1,
9,
3,
4,
4,
6,
4,
2,
1,
8,
2,
5,
4,
```

```
8,
2,
4,
0,
0,
2,
3,
2,
7,
7,
0,
8,
7,
4,
4,
7,
9,
6,
9,
0,
9,
8,
0,
4,
6,
0,
6,
3,
5,
4,
8,
3,
3,
9,
3,
3,
2,
7,
8,
0,
8,
8,
1,
7,
0,
6,
5,
4,
3,
```

```
3,
0,
9,
6,
3,
8,
0,
9,
9,
6,
2,
6,
8,
5,
7,
8,
6,
0,
2,
4,
0,
2,
2,
3,
1,
9,
7,
5,
1,
0,
8,
4,
6,
2,
6,
7,
7,
3,
2,
9,
8,
2,
2,
9,
2,
7,
3,
5,
9,
```

```
1,
8,
0,
2,
0,
5,
6,
1,
3,
7,
6,
7,
1,
2,
5,
8,
0,
3,
7,
8,
4,
0,
9,
1,
8,
6,
7,
7,
4,
3,
4,
9,
1,
4,
5,
1,
7,
3,
9,
7,
6,
9,
1,
3,
7,
8,
3,
3,
6,
```

```
4,
2,
4,
5,
8,
5,
1,
1,
4,
4,
3,
1,
0,
7,
7,
0,
7,
9,
4,
4,
8,
5,
5,
4,
0,
8,
2,
1,
0,
8,
4,
8,
0,
4,
0,
6,
1,
7,
3,
2,
6,
7,
2,
6,
9,
3,
1,
4,
6,
```

```
2,
5,
9,
2,
0,
6,
2,
1,
7,
3,
4,
1,
0,
5,
4,
3,
1,
1,
7,
4,
9,
9,
4,
8,
4,
0,
2,
4,
5,
1,
1,
6,
4,
7,
1,
9,
4,
2,
4,
1,
5,
5,
3,
8,
3,
1,
4,
5,
6,
```

```
8,
9,
4,
1,
5,
3,
8,
0,
3,
2,
5,
1,
2,
8,
3,
4,
4,
0,
8,
8,
3,
3,
1,
7,
3,
5,
9,
6,
3,
2,
6,
1,
3,
6,
0,
7,
2,
1,
7,
1,
4,
2,
4,
2,
1,
7,
9,
6,
1,
```

```
1,
2,
4,
3,
1,
7,
7,
4,
8,
0,
9,
3,
1,
3,
1,
0,
7,
7,
0,
3,
5,
5,
2,
7,
6,
6,
9,
2,
8,
3,
9,
2,
2,
5,
6,
0,
8,
2,
9,
2,
8,
2,
8,
8,
7,
4,
7,
3,
0,
```

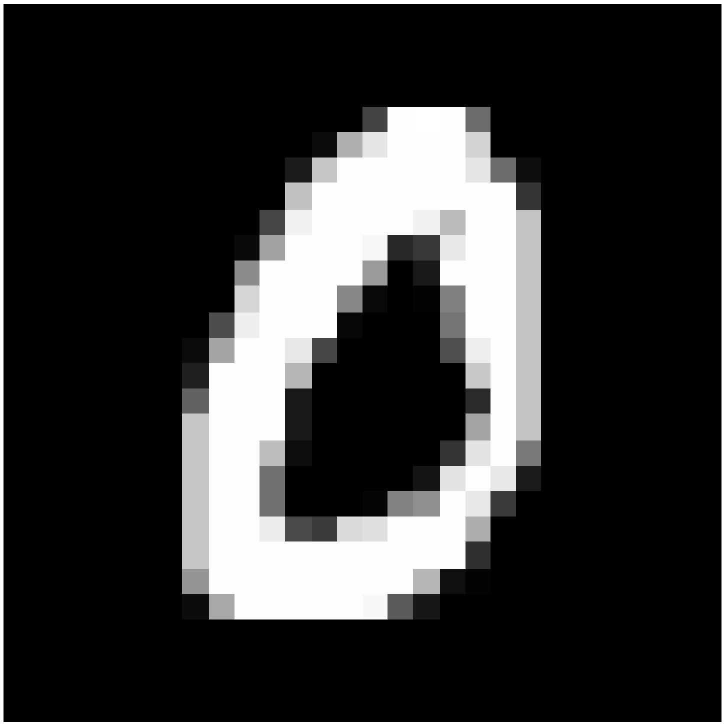
```
6,
6,
3,
2,
1,
3,
2,
2,
9,
3,
0,
0,
5,
7,
8,
1,
4,
4,
6,
0,
2,
9,
1,
4,
7,
4,
7,
3,
9,
8,
8,
4,
7,
1,
2,
1,
2,
2,
3,
2,
3,
2,
3,
9,
1,
7,
4,
0,
3,
```

```
5,
5,
8,
6,
3,
2,
6,
7,
6,
6,
3,
2,
7,
9,
1,
1,
7,
4,
6,
4,
9,
5,
2,
3,
3,
4,
7,
8,
9,
1,
1,
0,
9,
1,
4,
4,
5,
4,
0,
6,
2,
2,
3,
1,
5,
1,
2,
0,
3,
```

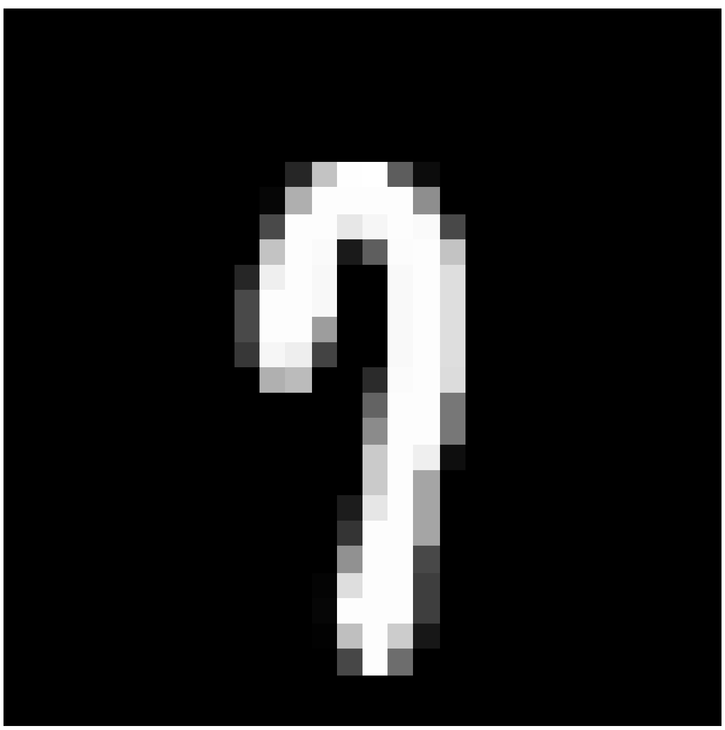
```
 8,
 1,
 2,
 6,
 7,
 1,
 6,
 2,
 3,
 9,
 0,
 1,
 2,
 2,
 0,
 8,
 9,
 ...]
```

```python
def visualize_image3(index, dataset='train'):
    if dataset == 'train':
        image_data = xtrain[index]
        label = ytrain[index]
    elif dataset == 'test':
        image_data = xtest[index]
        label = ypred3[index]
    else:
        print("Invalid dataset. Choose 'train' or 'test'.")
        return

    plt.imshow(image_data, cmap='gray')
    plt.title(f"Label: {label}")
    plt.axis('off')
    plt.show()

# Example usage:
visualize_image3(95, 'train')
visualize_image3(5600, 'test')
```

Label: 0

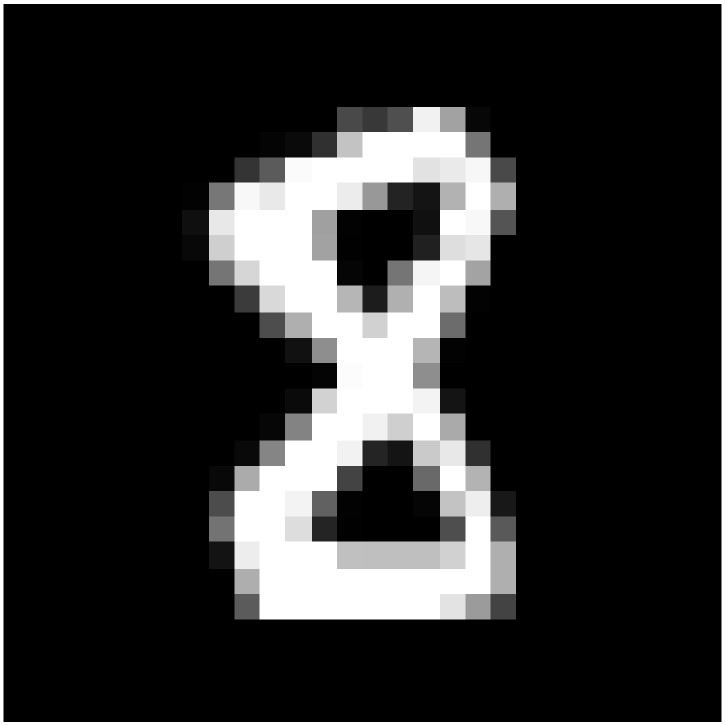


Label: 7



```
visualize_image3(179,'test')
```

Label: 8



```
visualize_image3(1290,'test')
```

Label: 3