```
In [1]: import tensorflow as tf
        from tensorflow.keras import datasets,layers,models
        import matplotlib.pyplot as pt
        import numpy as np
```

```
In [2]: (x_train,y_train),(x_test,y_test)=tf.keras.datasets.mnist.load_data()
```

```
In [3]: x_train.shape
```

```
Out[3]: (60000, 28, 28)
```

```
In [4]: y_test.shape
```

```
Out[4]: (10000,)
```

```
In [5]: x_train=x_train/255.0
        x_test=x_test/255.0
```

```
In [6]: cnn=models.Sequential([
            layers.Flatten(input_shape=(28,28)),
            layers.Dense(56,activation='relu'),
            layers.Dense(10,activation='softmax')
        ])
```

```
C:\Users\manej\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras\src\layers\reshaping\flatten.py:37: UserWarnin
g: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)`
object as the first layer in the model instead.
  super().__init__(**kwargs)
```

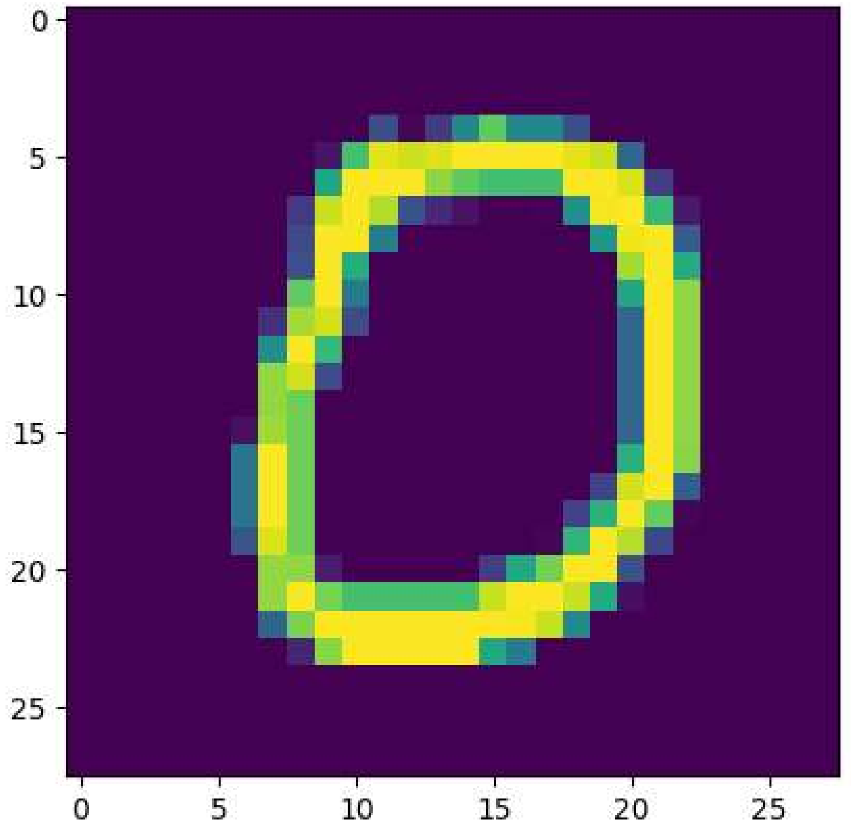
```
In [7]: cnn.compile(optimizer='SGD',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

```
In [8]: measures=cnn.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=5)
```

```
Epoch 1/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.7121 - loss: 1.0629 - val_accuracy: 0.8988 - val_loss: 0.3740
Epoch 2/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8976 - loss: 0.3669 - val_accuracy: 0.9140 - val_loss: 0.3089
Epoch 3/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9117 - loss: 0.3137 - val_accuracy: 0.9206 - val_loss: 0.2801
Epoch 4/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9180 - loss: 0.2852 - val_accuracy: 0.9274 - val_loss: 0.2574
Epoch 5/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9248 - loss: 0.2646 - val_accuracy: 0.9318 - val_loss: 0.2411
```

```
In [20]: pt.imshow(x_test[10])
```

```
Out[20]: <matplotlib.image.AxesImage at 0x23504ce5a30>
```



```
In [21]: pred=cnn.predict(x_test)
```

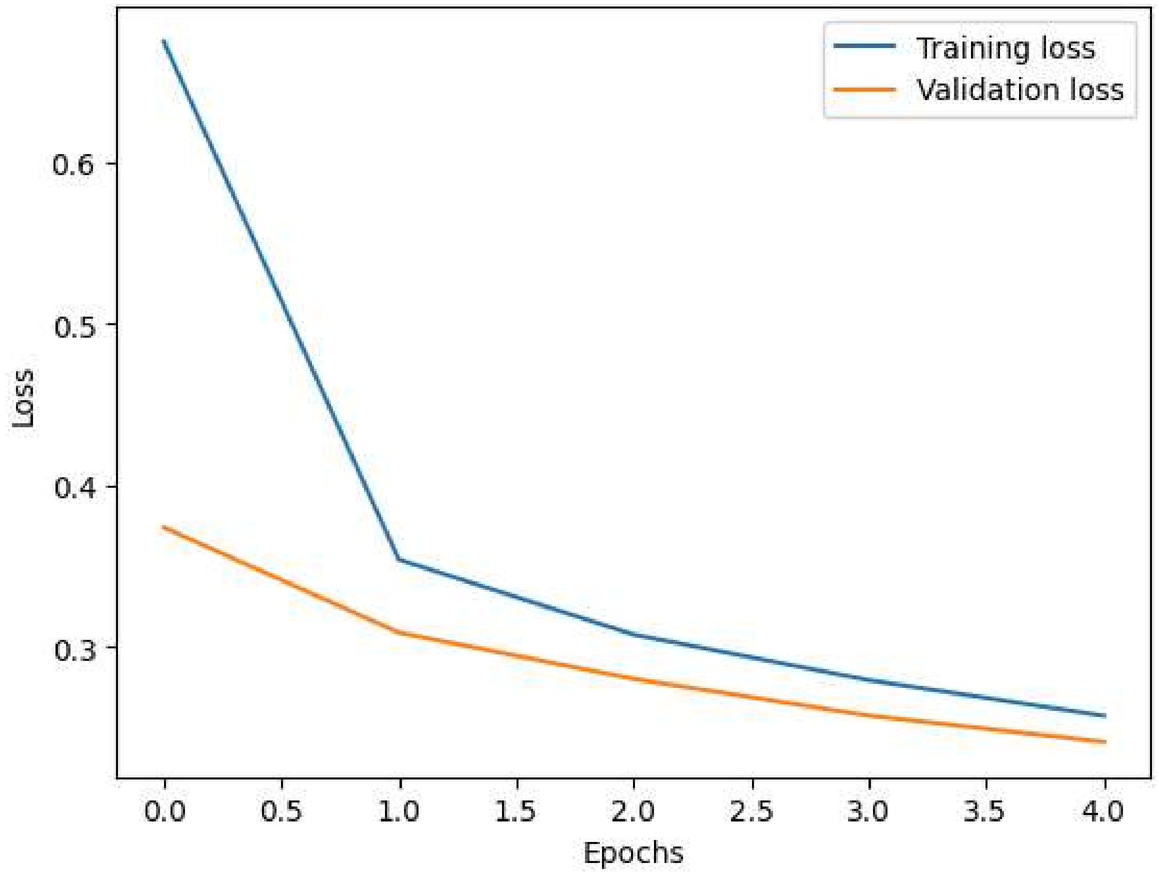
```
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 617us/step
```

```
In [23]: np.argmax(pred[10])
```

```
Out[23]: 0
```

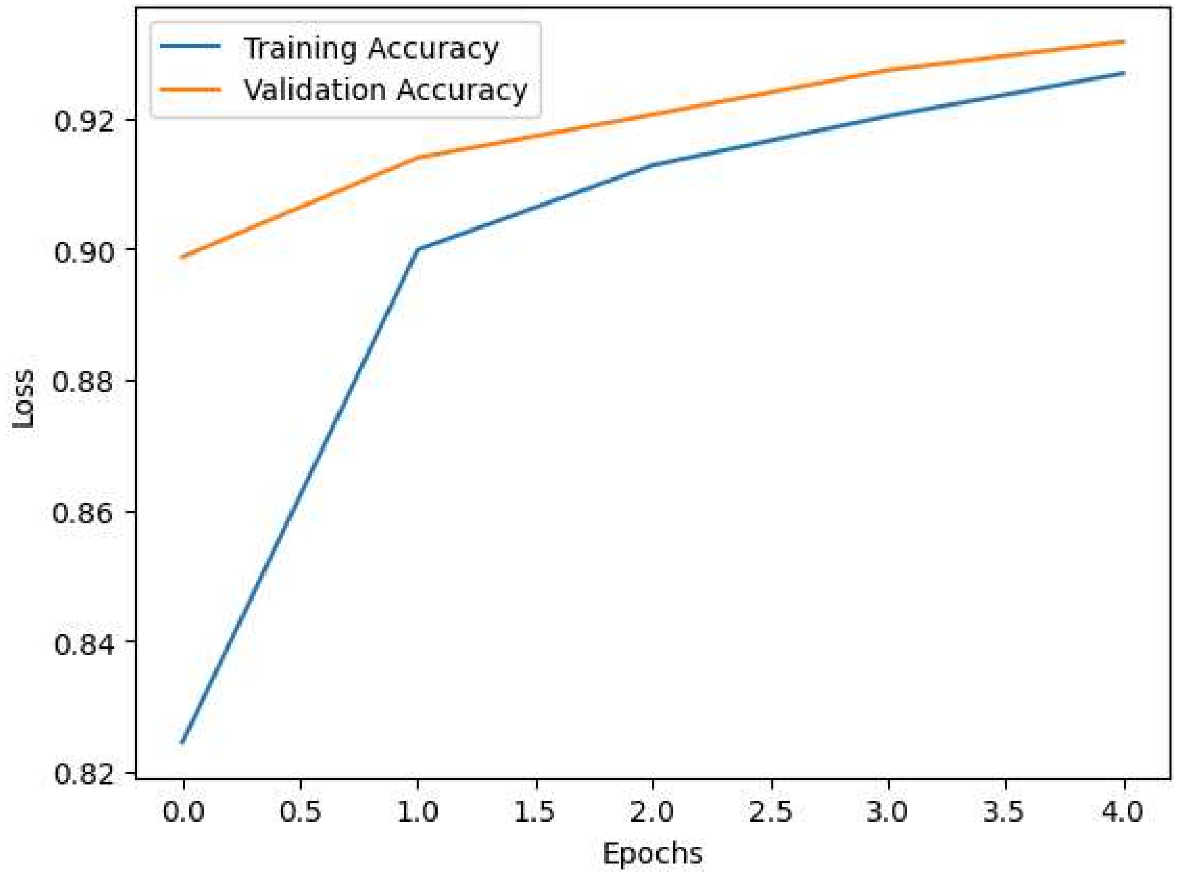
```
In [15]: pt.plot(measures.history['loss'],label="Training loss")
         pt.plot(measures.history['val_loss'],label="Validation loss")
         pt.xlabel("Epochs")
         pt.ylabel("Loss")
         pt.legend()
```

```
Out[15]: <matplotlib.legend.Legend at 0x235028755e0>
```



```
In [16]: pt.plot(measures.history['accuracy'],label='Training Accuracy')
         pt.plot(measures.history['val_accuracy'],label='Validation Accuracy')
         pt.xlabel("Epochs")
         pt.ylabel("Loss")
         pt.legend()
```

```
Out[16]: <matplotlib.legend.Legend at 0x23504bbe750>
```



```
In [ ]:
```