



```
In [1]: import tensorflow as tf
import keras
```

```
In [2]: from keras.models import Sequential
from keras.datasets import mnist
import matplotlib.pyplot as plt
import numpy as np
import random
```

```
In [3]: (X_train, y_train), (X_test, y_test) = mnist.load_data()
```

```
In [4]: keras.__version__
```

```
Out[4]: '3.11.3'
```

```
In [5]: tf.__version__
```

```
Out[5]: '2.16.1'
```

```
In [6]: import sys
print(sys.version)
```

```
3.12.4 | packaged by Anaconda, Inc. | (main, Jun 18 2024, 15:03:56) [MSC v.1929
64 bit (AMD64)]
```

```
In [7]: X_train = X_train/255
X_test = X_test/255
```

## Creating a model

```
In [15]: model = Sequential()
model.add(keras.layers.Flatten(input_shape=(28,28)))
model.add(keras.layers.Dense(128, activation="relu"))
model.add(keras.layers.Dense(64, activation="relu"))
model.add(keras.layers.Dense(10, activation="softmax"))
```

```
In [16]: model.compile(optimizer="sgd", loss="sparse_categorical_crossentropy", metrics
```

```
In [17]: model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=10)
```

```

Epoch 1/10
1875/1875 ————— 14s 7ms/step - accuracy: 0.8336 - loss: 0.6356 -
val_accuracy: 0.9103 - val_loss: 0.3210
Epoch 2/10
1875/1875 ————— 20s 7ms/step - accuracy: 0.9156 - loss: 0.2990 -
val_accuracy: 0.9271 - val_loss: 0.2607
Epoch 3/10
1875/1875 ————— 21s 7ms/step - accuracy: 0.9295 - loss: 0.2475 -
val_accuracy: 0.9380 - val_loss: 0.2204
Epoch 4/10
1875/1875 ————— 21s 7ms/step - accuracy: 0.9388 - loss: 0.2134 -
val_accuracy: 0.9432 - val_loss: 0.2002
Epoch 5/10
1875/1875 ————— 20s 7ms/step - accuracy: 0.9461 - loss: 0.1872 -
val_accuracy: 0.9487 - val_loss: 0.1761
Epoch 6/10
1875/1875 ————— 13s 7ms/step - accuracy: 0.9530 - loss: 0.1664 -
val_accuracy: 0.9532 - val_loss: 0.1619
Epoch 7/10
1875/1875 ————— 20s 7ms/step - accuracy: 0.9568 - loss: 0.1500 -
val_accuracy: 0.9569 - val_loss: 0.1457
Epoch 8/10
1875/1875 ————— 21s 7ms/step - accuracy: 0.9610 - loss: 0.1367 -
val_accuracy: 0.9580 - val_loss: 0.1386
Epoch 9/10
1875/1875 ————— 20s 7ms/step - accuracy: 0.9645 - loss: 0.1252 -
val_accuracy: 0.9612 - val_loss: 0.1284
Epoch 10/10
1875/1875 ————— 20s 7ms/step - accuracy: 0.9670 - loss: 0.1152 -
val_accuracy: 0.9644 - val_loss: 0.1230

```

Out[17]: <keras.src.callbacks.history.History at 0x2011ca0dbe0>

## Evaluate the model

```

In [18]: test_loss, test_acc = model.evaluate(X_test, y_test)
          print(f"Loss: {test_loss}")
          print(f"Acc: {test_acc}")

```

```

313/313 ————— 2s 5ms/step - accuracy: 0.9644 - loss: 0.1230
Loss: 0.12300097942352295
Acc: 0.9643999934196472

```

```

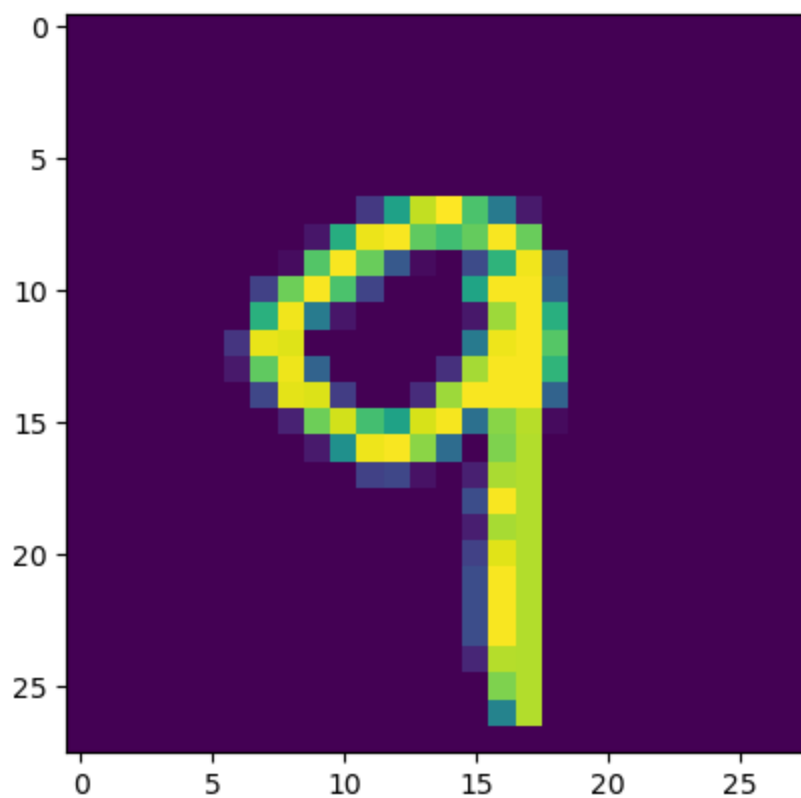
In [19]: n = random.randint(0, 999)

```

```

In [20]: plt.imshow(X_test[n])
          plt.show()

```



```
In [21]: prediction = model.predict(X_test)
         print(f"Digit: {np.argmax(prediction[n])}")
```

313/313 ————— 1s 4ms/step  
Digit: 9

In [ ]: