To recognize handwritten digits using deep learning in Python, you typically use libraries like TensorFlow or PyTorch along with the MNIST dataset, which contains 70,000 images of handwritten digits (0–9). Here's a simple yet effective example using **TensorFlow (Keras API)** to get you started:

### **Handwritten Digit Recognition with TensorFlow/Keras**

import tensorflow as tf

from tensorflow.keras.datasets import mnist

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Flatten

from tensorflow.keras.utils import to\_categorical

# Step 1: Load the MNIST dataset

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

# Step 2: Normalize the data

x\_train, x\_test = x\_train / 255.0, x\_test / 255.0

# Step 3: One-hot encode the labels

y\_train = to\_categorical(y\_train)

y\_test = to\_categorical(y\_test)

# Step 4: Build the deep learning model

model = Sequential([

Flatten(input\_shape=(28, 28)),

Dense(128, activation='relu'),

Dense(64, activation='relu'),

Dense(10, activation='softmax') # 10 classes for digits 0–9

])

# Step 5: Compile the model

model.compile(optimizer='adam',

loss='categorical\_crossentropy',

metrics=['accuracy'])

# Step 6: Train the model

model.fit(x\_train, y\_train, epochs=5, batch\_size=32, validation\_split=0.1)

# Step 7: Evaluate the model

test\_loss, test\_acc = model.evaluate(x\_test, y\_test)

print(f'Test accuracy: {test\_acc:.4f}')

# Optional: Predict a digit

import matplotlib.pyplot as plt

import numpy as np

# Pick a sample to predict

index = 0

sample = x\_test[index].reshape(1, 28, 28)

prediction = model.predict(sample)

predicted\_digit = np.argmax(prediction)

# Show result

plt.imshow(x\_test[index], cmap='gray')

plt.title(f'Predicted Digit: {predicted\_digit}')

plt.show()