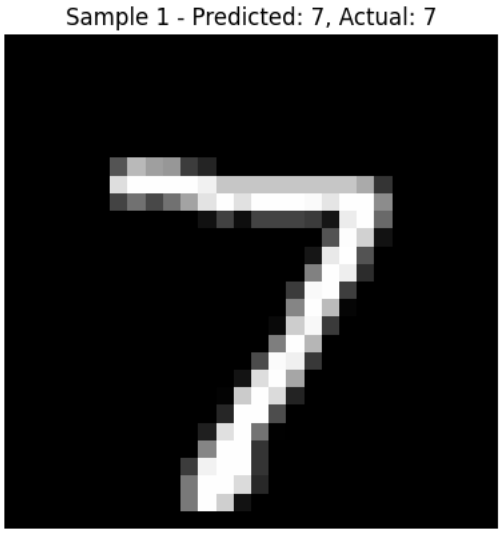
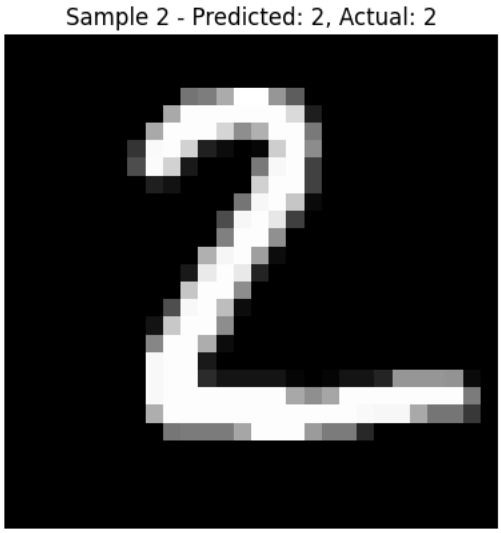
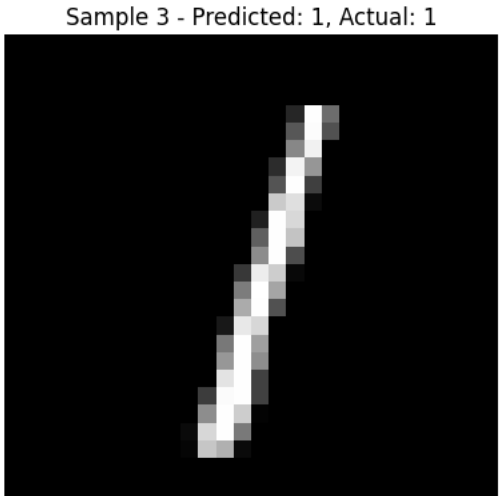
Double-click (or enter) to edit

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping

np.random.seed(42)
tf.random.set_seed(42)

feature_vector_length = 784
num_classes = 10

(X_train, Y_train), (X_test, Y_test) = mnist.load_data()

X_train = X_train.reshape(X_train.shape[0], feature_vector_length).astype('float32') / 255
X_test = X_test.reshape(X_test.shape[0], feature_vector_length).astype('float32') / 255

Y_train = to_categorical(Y_train, num_classes)
Y_test = to_categorical(Y_test, num_classes)

input_shape = (feature_vector_length,)

model = Sequential()
model.add(Dense(350, input_shape=input_shape, activation='relu'))
model.add(Dense(50, activation='relu'))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

early_stop = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)

model.fit(X_train, Y_train, epochs=10, batch_size=250, verbose=1, validation_split=0.2, callbacks=[early_stop])

test_loss, test_accuracy = model.evaluate(X_test, Y_test, verbose=1)
print(f'Test results - Loss: {test_loss:.4f} - Accuracy: {test_accuracy:.4f}')

predictions = model.predict(X_test[:5])
predicted_classes = np.argmax(predictions, axis=1)
true_classes = np.argmax(Y_test[:5], axis=1)

for i in range(5):
    plt.imshow(X_test[i].reshape(28, 28), cmap='gray')
    plt.title(f"Sample {i+1} - Predicted: {predicted_classes[i]}, Actual: {true_classes[i]}")
    plt.axis('off')
    plt.show()
```

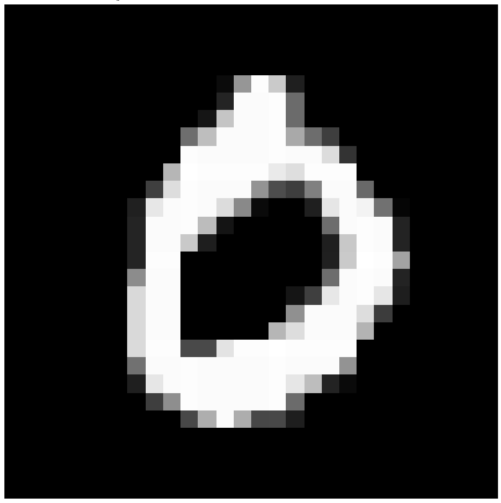
```
Epoch 1/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 3s 8ms/step - accuracy: 0.8060 - loss: 0.7126 - val_accuracy: 0.9487 - val_loss: 0.1745
Epoch 2/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 2s 7ms/step - accuracy: 0.9541 - loss: 0.1578 - val_accuracy: 0.9616 - val_loss: 0.1285
Epoch 3/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - accuracy: 0.9702 - loss: 0.1027 - val_accuracy: 0.9689 - val_loss: 0.1087
Epoch 4/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 3s 8ms/step - accuracy: 0.9792 - loss: 0.0742 - val_accuracy: 0.9712 - val_loss: 0.0998
Epoch 5/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 3s 10ms/step - accuracy: 0.9842 - loss: 0.0555 - val_accuracy: 0.9728 - val_loss: 0.0933
Epoch 6/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - accuracy: 0.9884 - loss: 0.0411 - val_accuracy: 0.9731 - val_loss: 0.0925
Epoch 7/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step - accuracy: 0.9920 - loss: 0.0304 - val_accuracy: 0.9734 - val_loss: 0.0946
Epoch 8/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - accuracy: 0.9950 - loss: 0.0232 - val_accuracy: 0.9732 - val_loss: 0.0986
Epoch 9/10
192/192 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - accuracy: 0.9966 - loss: 0.0183 - val_accuracy: 0.9750 - val_loss: 0.0934
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.9720 - loss: 0.0970
Test results - Loss: 0.0829 - Accuracy: 0.9760
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 50ms/step
```



Sample 4 - Predicted: 0, Actual: 0



Sample 5 - Predicted: 4, Actual: 4