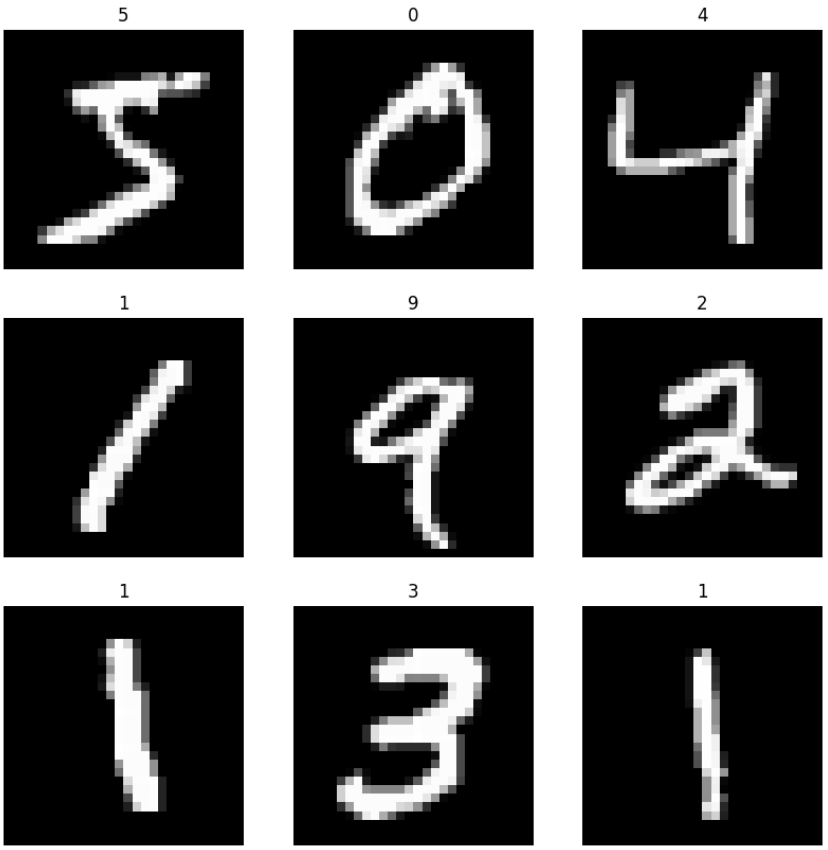
```
import tensorflow as tf
import matplotlib.pyplot as plt
# Load the MNIST dataset
mnist = tf.keras.datasets.mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images.shape
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 2s 0us/step
(60000, 28, 28)
```

```
# Create a subplot
plt.figure(figsize=(10, 10))
for i in range(9):
 # Display the image
 ax = plt.subplot(3, 3, i + 1)
 plt.imshow(train_images[i], cmap='gray')
 # Print the label
 plt.title(train_labels[i])
 plt.axis('off')
# Show the subplot
plt.show()
```




```
import tensorflow as tf
# Define the model
model = tf.keras.Sequential([
 tf.keras.layers.Flatten(input_shape=(28, 28)),
 tf.keras.layers.Dense(128, activation='relu'),
 tf.keras.layers.Dense(10, activation='softmax')
])
```

```
# Compile the model
model.compile(tf.keras.optimizers.Adam(learning_rate=0.001), loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
model.fit(train_images, train_labels, epochs=3)
```

```
Epoch 1/3
1875/1875 [==============================] - 10s 2ms/step - loss: 2.3575 - accuracy: 0.8537
Epoch 2/3
1875/1875 [==============================] - 5s 3ms/step - loss: 0.3922 - accuracy: 0.9088
Epoch 3/3
1875/1875 [==============================] - 5s 3ms/step - loss: 0.2837 - accuracy: 0.9294
<keras.callbacks.History at 0x7efa0655af50>
```

```
# Evaluate the model
train_accuracy=model.evaluate(train_images , train_labels)
```

```
1875/1875 [==============================] - 5s 3ms/step - loss: 0.2320 - accuracy: 0.9403
```

```
# Evaluate the model
test_accuracy=model.evaluate(test_images , test_labels)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.3066 - accuracy: 0.9318
```

```
print("train_accuracy",train_accuracy)
print("test_accuracy",test_accuracy)
```

```
train_accuracy [0.2320101112127304, 0.9403499960899353]
test_accuracy [0.3066263794898987, 0.9318000078201294]
```

```
import numpy as np
test_images_9 = test_images[:9]
# Create a subplot
fig, axes = plt.subplots(3, 3, figsize=(10, 10))
for i in range(9):
 test_images_9_1 = np.reshape(test_images_9[i], (1, 28, 28))
 prediction = model.predict(test_images_9_1)
 axes[i // 3, i % 3] = plt.subplot(3, 3, i + 1)
 axes[i // 3, i % 3].imshow(test_images_9[i])
 axes[i // 3, i % 3].set_title(f"Predicted: {prediction.argmax()}")
 axes[i // 3, i % 3].set_ylabel(f"Actual: {test_labels[i]}")
plt.show()
```

```
1/1 [==============================] - 0s 79ms/step
1/1 [==============================] - 0s 20ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 18ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 21ms/step
```