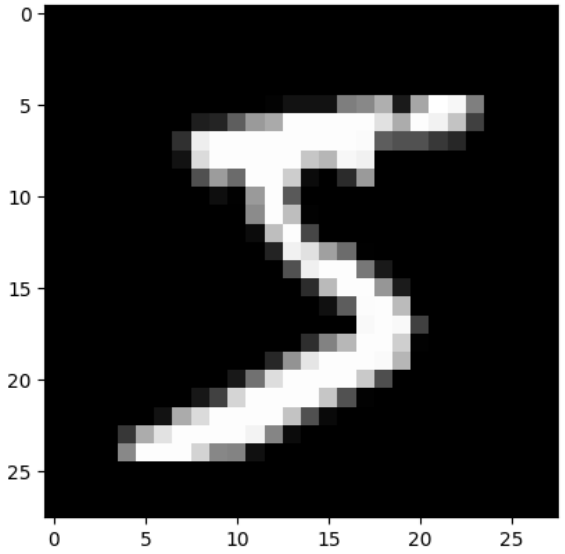
```
#import libraries
import tensorflow as tf
import matplotlib.pyplot as plt
```

```
# Load and split the data
(train_images, train_labels), (test_images ,test_labels) = tf.keras.datasets.mnist.load_data()
```

```
# Display sample data
print("Sample Data Shape:", train_images[0].shape)
plt.imshow(train_images[0], cmap='gray')
plt.show()
```

```
Sample Data Shape: (28, 28)
```




```
# Normalize the pixel values
train_images = train_images / 255.0
test_images = test_images / 255.0
```

```
# Print data shapes
print("Train Data Shape:", train_images.shape)
print("Test Data Shape:", test_images.shape)
# Print label shapes
print("Train Data Shape:", train_labels.shape)
print("Test Data Shape:", test_labels.shape)
```

```
Train Data Shape: (60000, 28, 28)
Test Data Shape: (10000, 28, 28)
Train Data Shape: (60000,)
Test Data Shape: (10000,)
```

```
# define neural network model

# model = tf.keras.models.Sequential()
# model.add(tf.keras.layers.Flatten(input_shape=(28,28)))
# model.add(tf.keras.layers.Dense(128,activation='relu'))
# model.add(tf.keras.layers.Dense(10,activation='softmax'))

model = tf.keras.models.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(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

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

```
Epoch 1/3
1875/1875 [==============================] - 14s 7ms/step - loss: 0.2546 - accuracy: 0.9275
Epoch 2/3
1875/1875 [==============================] - 9s 5ms/step - loss: 0.1115 - accuracy: 0.9672
Epoch 3/3
1875/1875 [==============================] - 11s 6ms/step - loss: 0.0777 - accuracy: 0.9765
<keras.src.callbacks.History at 0x79c08fc9a080>
```

```
# Evaluate the model
test_loss, test_accuracy = model.evaluate(test_images, test_labels)
print("Test Accuracy:", test_accuracy)
```

```
313/313 [==============================] - 1s 3ms/step - loss: 0.0871 - accuracy: 0.9726
Test Accuracy: 0.972599983215332
```

```
# Save the model
model.save("mnist_model")
```

```
# Load the model
new_model = tf.keras.models.load_model("mnist_model")
```

```
# Evaluate the loaded model
new_test_loss, new_test_accuracy = new_model.evaluate(test_images, test_labels)
print("Loaded Model Test Accuracy:", new_test_accuracy)
```

```
313/313 [==============================] - 2s 4ms/step - loss: 0.0871 - accuracy: 0.9726
Loaded Model Test Accuracy: 0.972599983215332
```