```
import tensorflow as tf
```

```
mnist = tf.keras.datasets.fashion_mnist
```

```
(training_images, training_labels),(test_images,test_labels)=mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
32768/29515 [=================================] - 0s 0us/step
40960/29515 [==========================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
26427392/26421880 [==============================] - 0s 0us/step
26435584/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-
16384/5148 [==================================================================================
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-
4423680/4422102 [==============================] - 0s 0us/step
4431872/4422102 [==============================] - 0s 0us/step
```

```
import numpy as np
np.set_printoptions(linewidth=200)
import matplotlib.pyplot as plt
plt.imshow(training_images[0])
print(training_labels[0])
print(training_images[0])
```

```
9
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0  13  73   0   0   1   4
 [  0   0   0   0   0   0   0   0   0   0   0   0   3   0  36 136 127  62  54   0   0
 [  0   0   0   0   0   0   0   0   0   0   0   0   6   0 102 204 176 134 144 123  23
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 155 236 207 178 107 156 161 16
 [  0   0   0   0   0   0   0   0   0   0   0   1   0  69 207 223 218 216 216 163 127 12
 [  0   0   0   0   0   0   0   0   0   1   1   1   0 200 232 232 233 229 223 223 215 23
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 183 225 216 223 228 235 227 224 22
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 193 228 218 213 198 180 212 210 23
 [  0   0   0   0   0   0   0   0   0   1   3   0  12 219 220 212 218 192 169 227 208 23
 [  0   0   0   0   0   0   0   0   0   0   6   0  99 244 222 220 218 203 198 221 215 23
 [  0   0   0   0   0   0   0   0   0   4   0   0  55 236 228 230 228 240 232 213 218 22
 [  0   0   1   4   6   7   2   0   0   0   0   0 237 226 217 223 222 219 222 221 216 22
 [  0   3   0   0   0   0   0   0   0  62 145 204 228 207 213 221 218 208 211 218 224 22
 [  0   0   0   0  18  44  82 107 189 228 220 222 217 226 200 205 211 230 224 234 176 18
 [  0  57 187 208 224 221 224 208 204 214 208 209 200 159 245 193 206 223 255 255 221 23
 [  3 202 228 224 221 211 211 214 205 205 205 220 240  80 150 255 229 221 188 154 191 23
 [ 98 233 198 210 222 229 229 234 249 220 194 215 217 241  65  73 106 117 168 219 221 23
```

```
plt.imshow(training_images[34])
print(training_images[34])
print(training_labels[34])
```

```
[[  0   0   0   0   0   0   0   0   0  42 110   1   0   0   0   0   0 156  21   0   0
 [  0   0   0   0   0   1  32  56  40  21 195 170 173 193 183 183 243 135   3  40  44  4
 [  0   0   0   0   0  48  50  25   7   1   1  38 173 255 220 154  23   0   0   0  13  3
 [  0   0   0   0  25  38  25  19  13  17   0   0   0   0   0   0   0   7  11  11  13  1
 [  0   0   0   5  65  25  46  27   9  11  13   3   0   0   0   5   3   0   0   5  15  2
 [  0   0   0  28  46  42  32  46   1   0   0   0   0   0   3   0   1  27   9   1  13  2
 [  0   0   0  59  25  50  54   9  94 160  98  75  81 108 131  92  61  79 129 148  63  4
 [  0   0   0  15  40  46  75  44  32  32   1 127 133  25 173  46 137 110   5   7  50  6
 [  0   0   0   0  13  52  81  59  36   0   0  32 106   7  86  23  13  11   0  13  67 12
 [  0   0   0   0   0  25 123  96  77  42  32  13   1   5   1  13  27  28  21  57  56  4
 [  0   0   0   0   0   0   0  79 112  73  59  44  27  32  32  48  48  46  54  67  34
 [  0   0   0   0   0   0   0  56 119  54  44  46  40  36  36  36  38  36  65  71  21
 [  0   0   0   0   0   0   0  36 112  50  44  46  40  42  42  40  38  36  52  90  21
 [  0   0   0   0   0   1   0  23 102  54  50  42  32  38  42  42  42  36  38  96  17
```

```
training_images= training_images/255.0
test_images= test_images/255.0
```

```
[  0   0   0   0   0   0   0  48  75  54  42  50  50  44  44  44  44  44  50  48  69
```

```
model = tf.keras.models.Sequential([tf.keras.layers.Flatten(),
                                    tf.keras.layers.Dense(128,activation=tf.nn.relu),
                                    tf.keras.layers.Dense(10,activation=tf.nn.softmax)])
```

```
[  0   0   0   0   0   0  25  71  40  52  56  46  48  48  50  42  48  52  44  44  46  5
```

```
model.compile(optimizer = tf.optimizers.Adam(),
              loss = 'sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(training_images, training_labels, epochs=95)
```

```
Epoch 68/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0773 - accuracy: 0
Epoch 69/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0746 - accuracy: 0
Epoch 70/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0761 - accuracy: 0

Epoch 71/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0738 - accuracy: 0
Epoch 72/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0730 - accuracy: 0
Epoch 73/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0738 - accuracy: 0
Epoch 74/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0705 - accuracy: 0
Epoch 75/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0724 - accuracy: 0
Epoch 76/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0704 - accuracy: 0
Epoch 77/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0711 - accuracy: 0
Epoch 78/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0685 - accuracy: 0
Epoch 79/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0674 - accuracy: 0
Epoch 80/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0655 - accuracy: 0
```

```
Epoch 81/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0661 - accuracy: 0
Epoch 82/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0673 - accuracy: 0
Epoch 83/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0632 - accuracy: 0
Epoch 84/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0629 - accuracy: 0
Epoch 85/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0636 - accuracy: 0
Epoch 86/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0618 - accuracy: 0
Epoch 87/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0628 - accuracy: 0
Epoch 88/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0604 - accuracy: 0
Epoch 89/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0599 - accuracy: 0
Epoch 90/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0617 - accuracy: 0
Epoch 91/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0587 - accuracy: 0
Epoch 92/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0573 - accuracy: 0
Epoch 93/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0589 - accuracy: 0
Epoch 94/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0579 - accuracy: 0
Epoch 95/95
1875/1875 [==============================] - 4s 2ms/step - loss: 0.0551 - accuracy: 0
<keras.callbacks.History at 0x7f614cc15e10>
```

```
model.evaluate(test_images,test_labels)
```

```
313/313 [==============================] - 1s 1ms/step - loss: 0.7170 - accuracy: 0.8936
[0.717032253742218, 0.8930000066757202]
```

```
mnist = tf.keras.datasets.mnist
(training_images, training_labels),(test_images,test_labels)= mnist.load_data()
training_images= training_images/255.0
test_images= test_images/255.0
model = tf.keras.Sequential([tf.keras.layers.Flatten(),
                             tf.keras.layers.Dense(1024, activation=tf.nn.relu),
                             tf.keras.layers.Dense(10, activation= tf.nn.softmax)])
model.compile(optimizer='adam', loss = 'sparse_categorical_crossentropy')
model.fit(training_images, training_labels,epochs=5)
model.evaluate(test_images, test_labels)
classifications = model.predict(test_images)
print(classifications[0])
print(test_labels[0])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
Epoch 1/5
1875/1875 [==============================] - 15s 8ms/step - loss: 0.1858
Epoch 2/5
1875/1875 [==============================] - 14s 8ms/step - loss: 0.0740
Epoch 3/5
1875/1875 [==============================] - 14s 8ms/step - loss: 0.0493
Epoch 4/5
1875/1875 [==============================] - 15s 8ms/step - loss: 0.0352
Epoch 5/5
1875/1875 [==============================] - 14s 8ms/step - loss: 0.0272
313/313 [==============================] - 1s 4ms/step - loss: 0.0818
[6.6295872e-11 2.1972890e-10 1.2483757e-08 1.5501132e-07 5.9972856e-12 2.8013349e-11 3.5
7
```

```
class myCallback(tf.keras.callbacks.Callback):
  def on_epoch_end(self, epoch, logs={}):
    if(logs.get('accuracy') >= 0.6): # Experiment with changing this value
      print("\nReached 60% accuracy so cancelling training!")
      self.model.stop_training = True

callbacks = myCallback()
mnist = tf.keras.datasets.fashion_mnist
(training_images, training_labels), (test_images, test_labels) = mnist.load_data()
training_images=training_images/255.0
test_images=test_images/255.0
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(512, activation=tf.nn.relu),
  tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(training_images, training_labels, epochs=5, callbacks=[callbacks])
```

```
Epoch 1/5
1875/1875 [==============================] - 9s 5ms/step - loss: 0.4740 - accuracy: 0.82

Reached 60% accuracy so cancelling training!
<keras.callbacks.History at 0x7f614550aad0>
```

✓ 9s completed at 3:51 PM