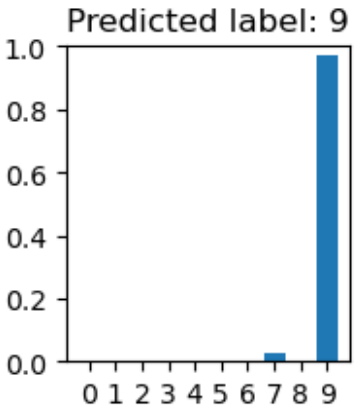
```
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt

#Load the dataset
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) =
fashion_mnist.load_data()

# Normalize the images
train_images = train_images / 255.0
test_images = test_images / 255.0


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

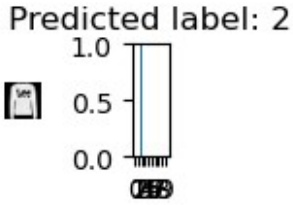
```
C:\Users\STES\anaconda3\Lib\site-packages\keras\src\layers\reshaping\
flatten.py:37: UserWarning: Do not pass an `input_shape`/`input_dim`
argument to a layer. When using Sequential models, prefer using an
`Input(shape)` object as the first layer in the model instead.
  super().__init__(**kwargs)
```

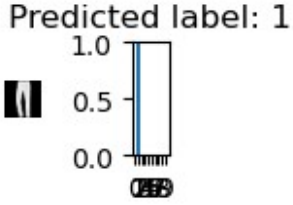
```
# Compile the model
model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])

# Train the model
model.fit(train_images, train_labels, epochs=10)
#Evaluate the model
test_loss, test_acc = model.evaluate(test_images, test_labels)
print('Test accuracy:', test_acc)
# Make predictions
predictions = model.predict(test_images)
predicted_labels =np.argmax(predictions, axis=1)
# Show some example images and their predicted labels
num_rows = 5
num_cols = 5
num_images = num_rows * num_cols
plt.figure(figsize=(2 * 2 *num_cols, 2 * num_rows))
for i in range(num_images):
    plt.subplot(num_rows, 2 * num_cols, 2 * i + 1)
    plt.imshow(test_images[i], cmap='gray')
    plt.axis('off')
    plt.subplot(num_rows, 2 * num_cols, 2 * i + 2)
    plt.bar(range(10),predictions[i])
    plt.xticks(range(10))
```

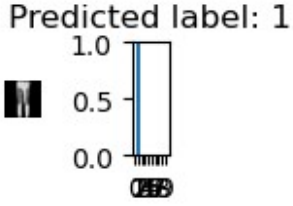
```
    plt.ylim([0, 1])
    plt.tight_layout()
    plt.title(f"Predicted label: {predicted_labels[i]}")
    plt.show()
```

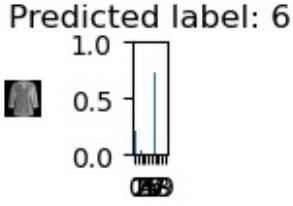
```
Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.7819 - loss:
0.6271
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8643 - loss:
0.3753
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8785 - loss:
0.3363
Epoch 4/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8881 - loss:
0.3088
Epoch 5/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8912 - loss:
0.2967
Epoch 6/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.8982 - loss:
0.2761
Epoch 7/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.9028 - loss:
0.2638
Epoch 8/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9054 - loss:
0.2564
Epoch 9/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9077 - loss:
0.2460
Epoch 10/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 2s 1ms/step - accuracy: 0.9128 - loss:
0.2349
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 617us/step - accuracy: 0.8900 - loss:
0.3190
Test accuracy: 0.8896999955177307
313/313 ━━━━━━━━━━━━━━━━━━━━ 0s 819us/step
```


Predicted label: 9
1.0
0.8
0.6
0.4
0.2
0.0
0 1 2 3 4 5 6 7 8 9

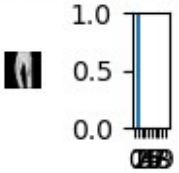

Predicted label: 2
1.0
0.5
0.0


Predicted label: 1
1.0
0.5
0.0

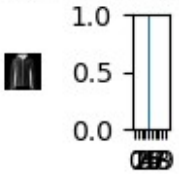

Predicted label: 1
1.0
0.5
0.0


Predicted label: 6
1.0
0.5
0.0

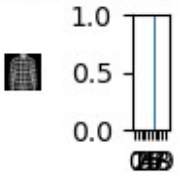
Predicted label: 1



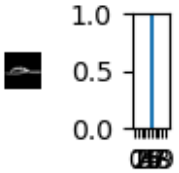
Predicted label: 4



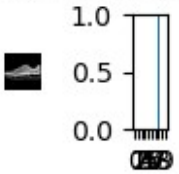
Predicted label: 6



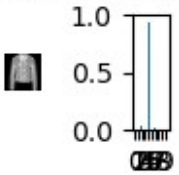
Predicted label: 5



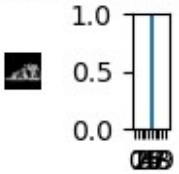
Predicted label: 7



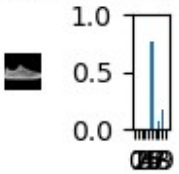
Predicted label: 4



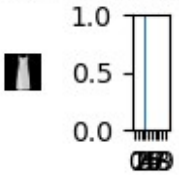
Predicted label: 5



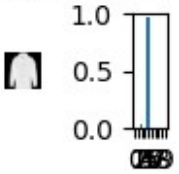
Predicted label: 5



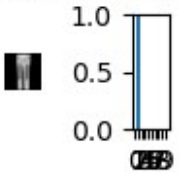
Predicted label: 3



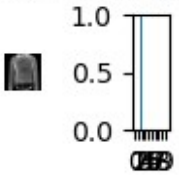
Predicted label: 4



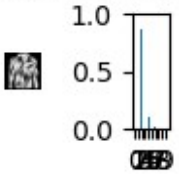
Predicted label: 1



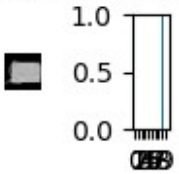
Predicted label: 2



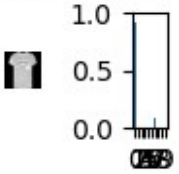
Predicted label: 2



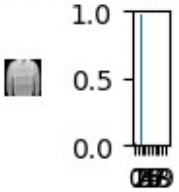
Predicted label: 8



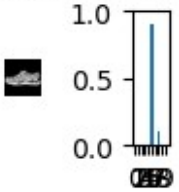
Predicted label: 0



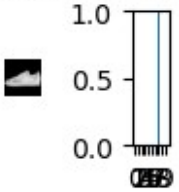
Predicted label: 2

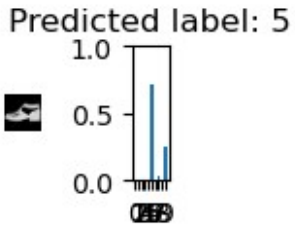


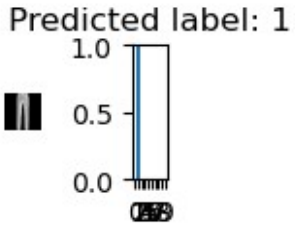
Predicted label: 5



Predicted label: 7




Predicted label: 5
1.0
0.5
0.0


Predicted label: 1
1.0
0.5
0.0