02_1_Fashion_MNIST

December 22, 2020

[1]:
```
# tensorflow tf.keras
import tensorflow as tf
from tensorflow import keras

# (helper)
import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)
```

```
2.4.0
```

[2]:
```
fashion_mnist = keras.datasets.fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.
 ↪load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/train-labels-idx1-ubyte.gz
32768/29515 [==================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/train-images-idx3-ubyte.gz
26427392/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/t10k-labels-idx1-ubyte.gz
8192/5148 [===============================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/t10k-images-idx3-ubyte.gz
4423680/4422102 [==============================] - 0s 0us/step
```

[3]:
```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
               'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

[4]:
```
train_images.shape
```

[4]:
```
(60000, 28, 28)
```

[5]:
```
len(train_labels)
```

[5]:
```
60000
```

[6]:
```
train_labels
```

[6]:
```
array([9, 0, 0, ..., 3, 0, 5], dtype=uint8)
```

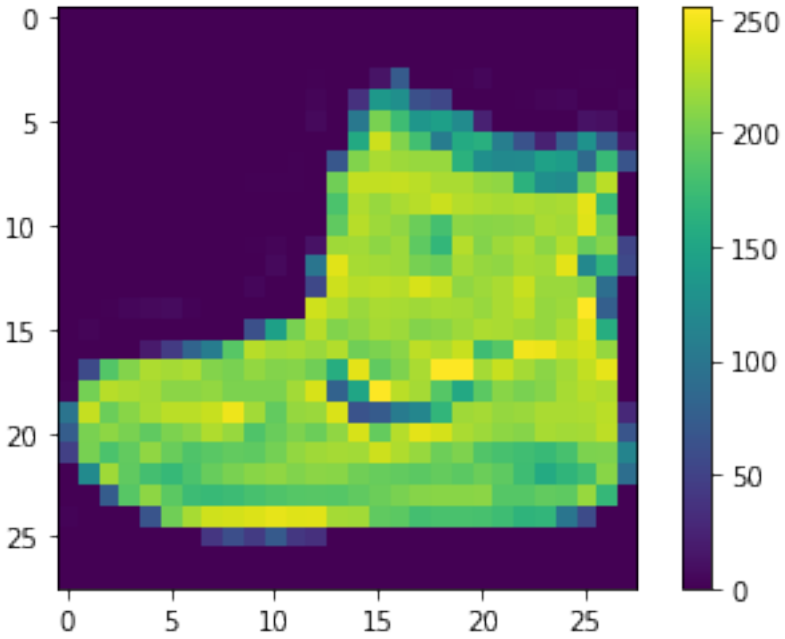
[7]:
```
test_images.shape
```

[7]:
```
(10000, 28, 28)
```

[8]:
```
len(test_labels)
```

[8]:
```
10000
```

[9]:
```
plt.figure()
plt.imshow(train_images[0])
plt.colorbar()
plt.grid(False)
plt.show()
```



[10]:
```
train_images = train_images / 255.0

test_images = test_images / 255.0
```

[11]:
```
plt.figure(figsize=(10,10))
for i in range(25):
    plt.subplot(5,5,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
```

```
    plt.imshow(train_images[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[train_labels[i]])
plt.show()
```

[12]:
```
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dense(10, activation='softmax')
])
```

[13]:
```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

[14]:
```
model.fit(train_images, train_labels, epochs=5)
```

```
Epoch 1/5
1875/1875 [==============================] - 4s 2ms/step - loss: 0.6434 -
accuracy: 0.7786
Epoch 2/5
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3822 -
accuracy: 0.8620
Epoch 3/5
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3408 -
accuracy: 0.8770
Epoch 4/5
1875/1875 [==============================] - 4s 2ms/step - loss: 0.3174 -
accuracy: 0.8823
Epoch 5/5
1875/1875 [==============================] - 4s 2ms/step - loss: 0.2944 -
accuracy: 0.8913
```

[14]:
```
<tensorflow.python.keras.callbacks.History at 0x7fd2cd25b7f0>
```

[15]:
```
test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2)

print('\n :', test_acc)
```

```
313/313 - 0s - loss: 0.3346 - accuracy: 0.8816

 : 0.881600022315979
```

[16]:
```
predictions = model.predict(test_images)
```

[17]:
```
predictions[0]
```

[17]:
```
array([6.2302977e-05, 2.3736226e-07, 1.1627841e-07, 4.2882613e-09,
       4.8808538e-08, 2.5991391e-02, 1.1044251e-06, 6.3801602e-02,
       3.6593701e-05, 9.1010660e-01], dtype=float32)
```

[18]:
```
np.argmax(predictions[0])
```

[18]:
```
9
```

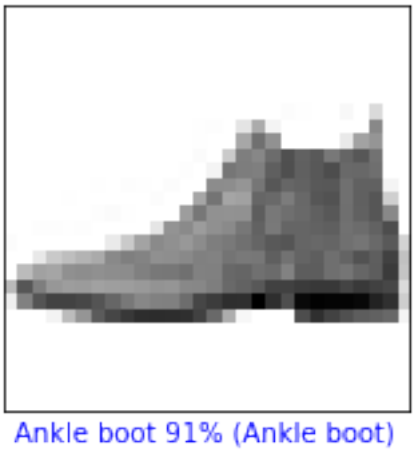
[19]:
```
test_labels[0]
```

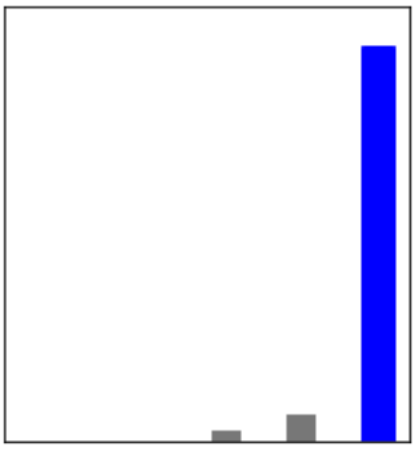
[19]:
```
9
```

[21]:
```
def plot_image(i, predictions_array, true_label, img):
  predictions_array, true_label, img = predictions_array[i], true_label[i],␣
 ↪img[i]
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])
```

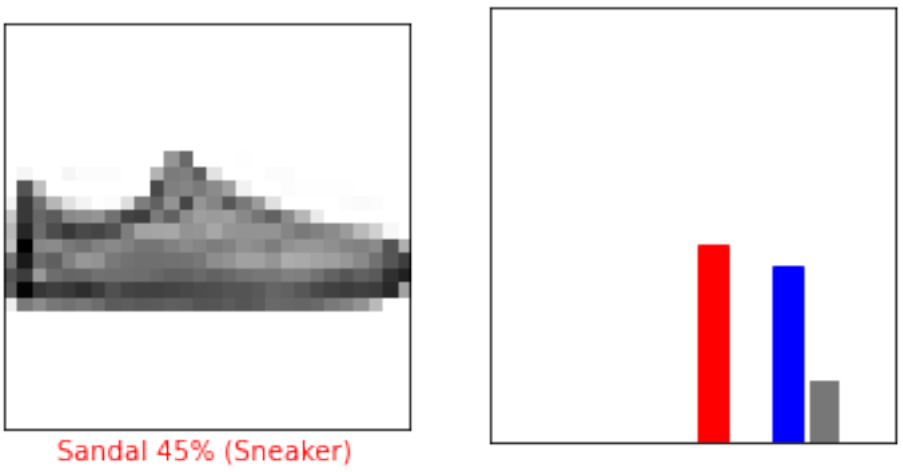
```
  plt.imshow(img, cmap=plt.cm.binary)

  predicted_label = np.argmax(predictions_array)
  if predicted_label == true_label:
    color = 'blue'
  else:
    color = 'red'

  plt.xlabel("{} {:2.0f}% ({})".format(class_names[predicted_label],
                                100*np.max(predictions_array),
                                class_names[true_label]),
                                color=color)

def plot_value_array(i, predictions_array, true_label):
  predictions_array, true_label = predictions_array[i], true_label[i]
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])
  thisplot = plt.bar(range(10), predictions_array, color="#777777")
  plt.ylim([0, 1])
  predicted_label = np.argmax(predictions_array)

  thisplot[predicted_label].set_color('red')
  thisplot[true_label].set_color('blue')
```

[22]:
```
i = 0
plt.figure(figsize=(6,3))
plt.subplot(1,2,1)
plot_image(i, predictions, test_labels, test_images)
plt.subplot(1,2,2)
plot_value_array(i, predictions,  test_labels)
plt.show()
```

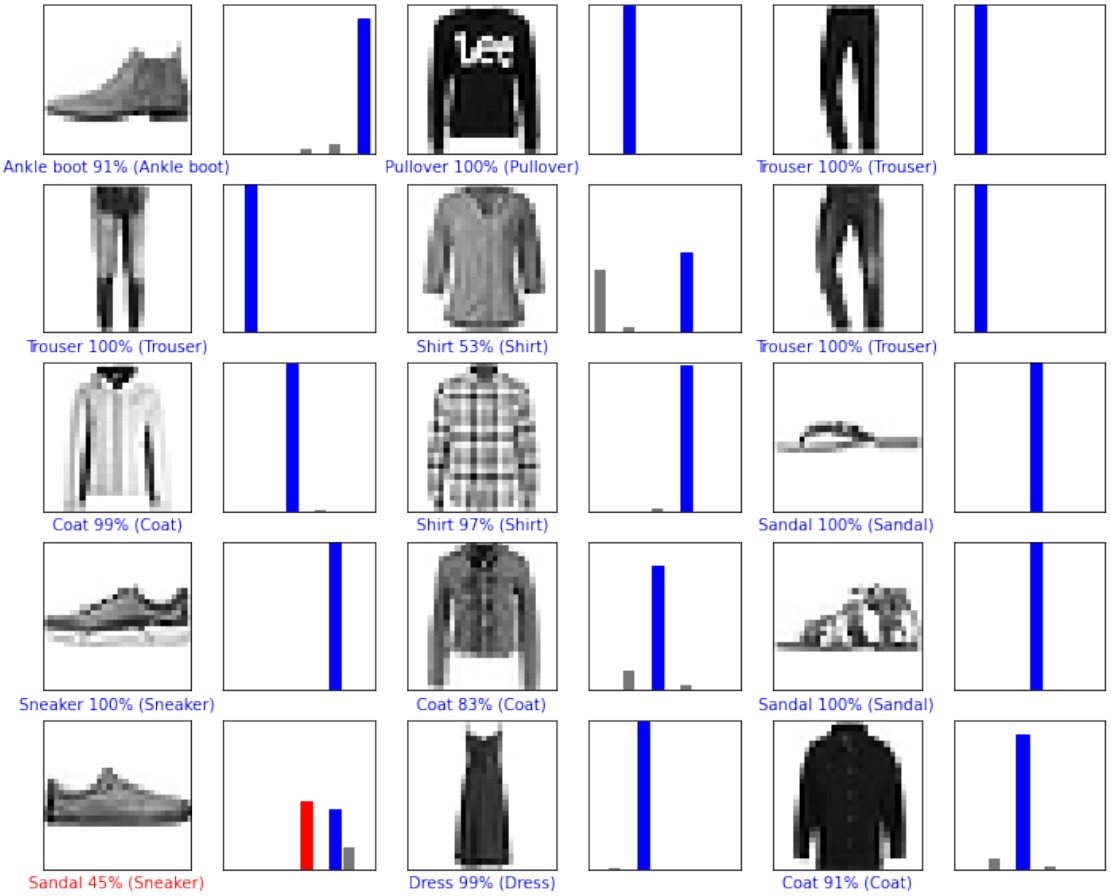




[23]:
```
i = 12
plt.figure(figsize=(6,3))
plt.subplot(1,2,1)
plot_image(i, predictions, test_labels, test_images)
plt.subplot(1,2,2)
plot_value_array(i, predictions,  test_labels)
plt.show()
```



[24]:
```
num_rows = 5
num_cols = 3
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)
  plot_image(i, predictions, test_labels, test_images)
  plt.subplot(num_rows, 2*num_cols, 2*i+2)
  plot_value_array(i, predictions, test_labels)
plt.show()
```



[25]:
```
img = test_images[0]

print(img.shape)
```

```
(28, 28)
```

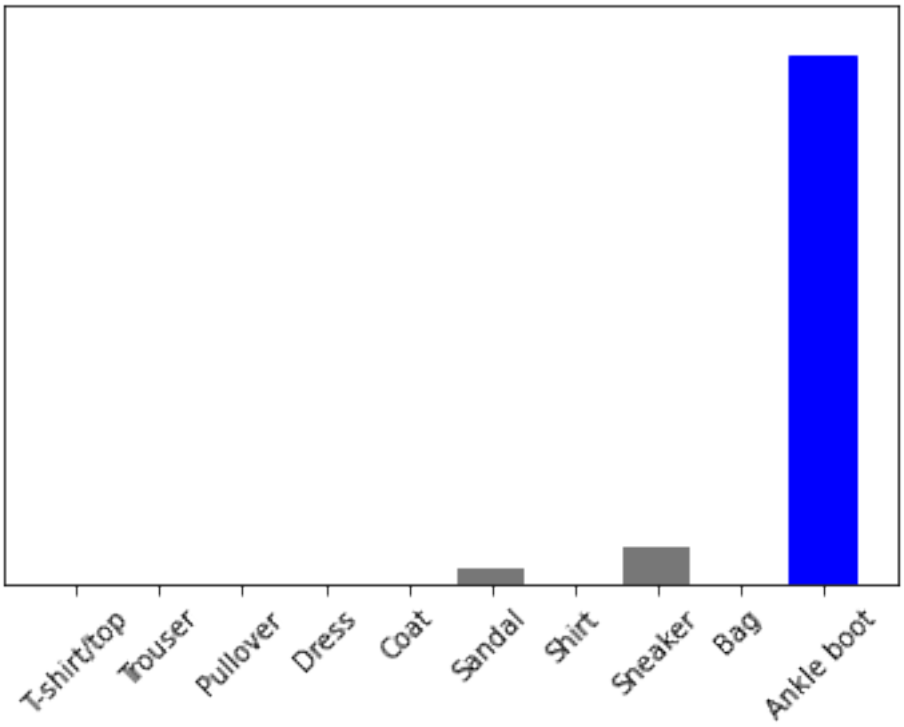
[26]:
```
img = (np.expand_dims(img,0))

print(img.shape)
```

```
(1, 28, 28)
```

[27]:
```
predictions_single = model.predict(img)

print(predictions_single)
```

```
[[6.2302861e-05 2.3736226e-07 1.1627829e-07 4.2882773e-09 4.8808634e-08
  2.5991429e-02 1.1044241e-06 6.3801624e-02 3.6593770e-05 9.1010660e-01]]
```

[28]:
```
plot_value_array(0, predictions_single, test_labels)
_ = plt.xticks(range(10), class_names, rotation=45)
```



[29]:
```
np.argmax(predictions_single[0])
```

[29]:
```
9
```