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

```
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
```

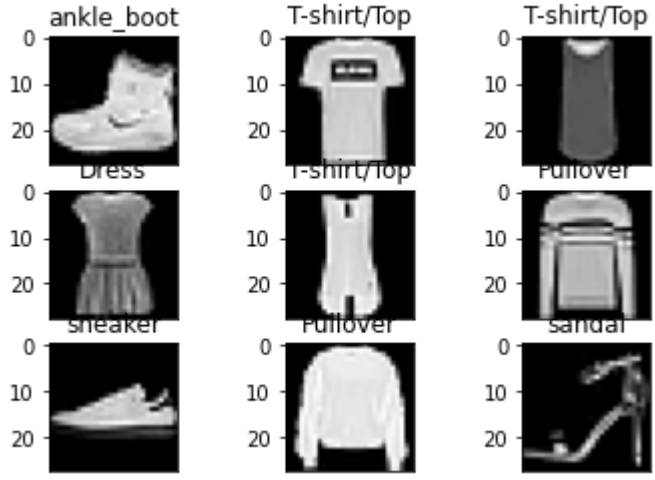
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-
32768/29515 [================================] - 0s 0us/step
40960/29515 [========================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-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

```
y_train
```

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

```
class_names = ['T-shirt/Top','Trouser','Pullover',
               'Dress','Coat','sandal','shirt',
               'sneaker','bag','ankle_boot']
```

```
for i in range(9):
  plt.subplot(3,3,i+1)
  plt.imshow(x_train[i], cmap='gray')
  plt.title(class_names[y_train[i]])
  plt.xticks([])
```



```
x_train.shape
```

```
(60000, 28, 28)
```

```
x_test.shape
```

```
(10000, 28, 28)
```

```
x_train = x_train / 255
x_test = x_test / 255
```

```
from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense,Flatten
```

```
model = Sequential([Flatten(input_shape=(28,28)),
                    Dense(128,activation='relu'),
                    Dense(10,activation='softmax')])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 128)               100480

 dense_1 (Dense)             (None, 10)                1290

=================================================================
Total params: 101,770
Trainable params: 101,770
Non-trainable params: 0
_________________________________________________________________
```

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

```
model.fit(x_train, y_train, epochs=10, batch_size=5)
```

```
Epoch 1/10
12000/12000 [==============================] - 32s 3ms/step - loss: 0.2487 - accuracy: 0
Epoch 2/10
12000/12000 [==============================] - 24s 2ms/step - loss: 0.2410 - accuracy: 0
Epoch 3/10
```

```
12000/12000 [==============================] - 25s 2ms/step - loss: 0.2360 - accuracy: 0
Epoch 4/10
12000/12000 [==============================] - 25s 2ms/step - loss: 0.2323 - accuracy: 0
Epoch 5/10
12000/12000 [==============================] - 24s 2ms/step - loss: 0.2273 - accuracy: 0
Epoch 6/10
12000/12000 [==============================] - 23s 2ms/step - loss: 0.2219 - accuracy: 0
Epoch 7/10
12000/12000 [==============================] - 23s 2ms/step - loss: 0.2168 - accuracy: 0
Epoch 8/10
12000/12000 [==============================] - 22s 2ms/step - loss: 0.2160 - accuracy: 0
Epoch 9/10
12000/12000 [==============================] - 23s 2ms/step - loss: 0.2088 - accuracy: 0
Epoch 10/10
12000/12000 [==============================] - 23s 2ms/step - loss: 0.2089 - accuracy: 0
<keras.callbacks.History at 0x7fa8f6f0aa10>
```

```
test_loss, test_acc = model.evaluate(x_test,y_test)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.4171 - accuracy: 0.8825
```

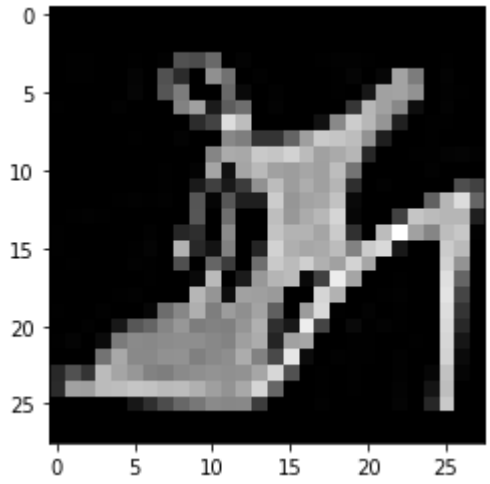
```
test_acc
```

```
0.8824999928474426
```

```
new = x_train[345]
```

```
plt.imshow(new, cmap="gray")
```

```
<matplotlib.image.AxesImage at 0x7fa8fd29e390>
```




```
predictions·=·model.predict(x_train)·
data=·np.argmax·(predictions·[345])·
class_names·[data]
```

```
'sandal'
```


✓ 0s completed at 10:48 AM